University of Macedonia
School of Information Sciences
Department of Applied Informatics

Recommendations in Mobile Commerce Environments: Supporting Quality and Privacy Requirements

Nikolaos Polatidis

PhD Thesis

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Thessaloniki, 7/2017

This Thesis Advisory Committee consists of the following members:

Supervisor: Associate Professor. Christos K. Georgiadis
(University of Macedonia, Thessaloniki, Greece)

Member: Senior Lecturer. Elias Pimenidis
(University of the West of England, Bristol, U.K)

Member: Associate Professor. Emmanouil Stiakakis
(University of Macedonia, Thessaloniki, Greece)

This Thesis Defense Committee consists of the following members:

Associate Professor. Christos K. Georgiadis
(University of Macedonia, Thessaloniki, Greece)

Senior Lecturer. Elias Pimenidis
(University of the West of England, Bristol, U.K)

Associate Professor. Emmanouil Stiakakis
(University of Macedonia, Thessaloniki, Greece)

Professor. Alexander Chatzigeorgiou
(University of Macedonia, Thessaloniki, Greece)

Professor. Ioannis Refanidis
(University of Macedonia, Thessaloniki, Greece)

Associate Professor. Apostolos N. Papadopoulos
(Aristotle University of Thessaloniki, Greece)

Professor. Haralambos Mouratidis
(University of Brighton, U.K)

Abstract

Recommender systems have become a technology that has been widely used by various online applications in situations where there is an information overload problem. Numerous applications such as e-commerce, movie recommendations, song recommendations and people to people recommendations in social networks helping the users, which has improved the quality of the experience and revenues of the vendor. The development of recommender systems has been focused mostly on the development of algorithms, such as collaborative filtering, that provide more accurate recommendations in web environments. However, the use of mobile devices and the rapid growth of the internet and networking infrastructure has brought the necessity of using mobile recommender systems in different environments and scenarios. In mobile environments, every recommendation system is developed with a specific task in mind since a mobile recommendation domain does not exist now. Furthermore, when a mobile recommender system is developed a set of components derived from traditional web recommender systems is used. However, little work has been done towards the identification of the requirements that have an impact on the development of mobile recommender systems.

In this thesis, the factors that affect the quality of recommender systems in mobile environments have been extensively analyzed by the author and novel work towards mobile recommender systems development is presented. To that end, the author assists in the development of mobile recommender systems by analyzing the factors that affect the recommendations in mobile scenarios and then makes four contributions: In the first contribution, the concept of static multi-level collaborative filtering for the improvement of the recommendations is introduced. In the second contribution, a dynamic multi-level collaborative filtering is proposed. In the third contribution, a method for privacy-preserving collaborative filtering recommendations is proposed, while in the fourth contribution a method for privacy-preserving context-aware mobile recommendations is presented.

The proposals have been tested on common experimental designs that consider real datasets from different recommendation domains, widely used evaluation metrics and comparisons to alternative methods where necessary. The results of the experiments

support the validity of the contributions and provide useful insights on their behavior, while supporting their practical applicability.

Keywords: Mobile Recommender Systems, Context, Collaborative Filtering, Privacy

Περίληψη

Τα συστήματα συστάσεων έχουν γίνει μια τεχνολογία που έχει χρησιμοποιηθεί ευρέως από διάφορες ηλεκτρονικές εφαρογές σε περιπτώσεις όπου υπάρχει πρόβλημα υπερφόρτωσης πληροφοριών. Πολυάριθμες εφαρμογές όπως το ηλεκτρονικό εμπόριο, οι συστάσεις ταινιών, οι συστάσεις τραγουδιού και οι συστάσεις ατόμων σε άτομα στα κοινωνικά δίκτυα βοηθούν τους χρήστες, γεγονός που βελτίωσε την ποιότητα της εμπειρίας των χρηστών αλλά και των εσόδων της εταιρείας που τα χρησιμοποιεί. Η ανάπτυξη συστημάτων συστάσεων έχει επικεντρωθεί κυρίως στην ανάπτυξη αλγορίθμων, όπως το συνεργατικό φιλτράρισμα, που παρέχουν πιο ακριβείς συστάσεις σε περιβάλλοντα web. Ωστόσο, η χρήση κινητών συσκευών και η ταχεία ανάπτυξη της υποδομής διαδικτύου και δικτύωσης έχουν φέρει την ανάγκη χρήσης συστημάτων συστάσεων σε κινητές συσκευές οι οποίες μπορεί να βρίσκονται σε εναλλασόμενα περιβάλλοντα. Σε κινητά περιβάλλοντα, κάθε σύστημα συστάσεων αναπτύσσεται με συγκεκριμένο στόχο δεδομένου ότι δεν υπάρχει εώς τώρα ένας τομέας συστάσεων για κινητά. Επιπλέον, όταν αναπτύσσεται ένα σύστημα συστάσεων σε κινητά περιβάλλοντα, χρησιμοποιείται ένα σύνολο τεχνολογιών που προέρχονται από τα συστήματα συστάσεων διαδικτυακού περιβάλλοντος. Μέχρι στιγμής, έχουν γίνει ελάχιστες προσπάθειες για τον εντοπισμό των απαιτήσεων που έχουν αντίκτυπο στην ανάπτυξη συστημάτων συστάσεων σε κινητά περιβάλλοντα.

Σε αυτή τη διατριβή, οι παράγοντες που επηρεάζουν την ποιότητα των συστημάτων συστάσεων σε κινητά περιβάλλοντα έχουν αναλυθεί εκτενώς από τον συγγραφέα και παρουσιάζονται νέες κατευθύνσεις προς την ανάπτυξη αυτών. Για το σκοπό αυτό, ο συγγραφέας βοηθά στην ανάπτυξη συστημάτων συστάσεων σε κινητά περιβάλλοντα αναλύοντας τους παράγοντες που επηρεάζουν τις συστάσεις σε κινητά σενάρια και στη συνέχεια κάνει τέσσερις συνεισφορές: Στην πρώτη συνεισφορά, η έννοια του στατικού συνεργατικού φιλτραρίσματος πολλαπλών επιπέδων παρουσιάζεται. Στη δεύτερη συνεισφορά προτείνεται ένα δυναμικό φιλτράρισμα πολλαπλών επιπέδων συνεργασίας. Στην τρίτη συνεισφορά προτείνεται μια μέθοδος για συστάσεις συνεργατικού φιλτραρίσματος που προστατεύει τα δεδομένα του χρήστη, ενώ στην τέταρτη συμβολή παρουσιάζεται μια μέθοδος για τη διαφύλαξη της προστασίας της ιδιωτικής ζωής όταν χρησιμοποιούνται μεταβλητές που είναι σχετικές με την τοποθεσία και τον περιβάλλοντα χώρο.

Οι συνεισφορές έχουν δοκιμαστεί εκτενώς εκτελώντας πειράματα με πραγματικά σύνολα δεδομένων από διαφορετικούς τομείς συστάσεων, ευρέως χρησιμοποιούμενες μετρήσεις αξιολόγησης και συγκρίσεις με εναλλακτικές μεθόδους. Τα αποτελέσματα των πειραμάτων υποστηρίζουν την εγκυρότητα των συμβολών και παρέχουν χρήσιμες γνώσεις σχετικά με τη συμπεριφορά τους, υποστηρίζοντας παράλληλα την πρακτική εφαρμογή τους.

Acknowledgements

This thesis is the result of personal hard work. However, this would have never been possible without the help of some people. I feel for the people that have helped me and I will try my best to express my gratitude.

At first, I would like to thank my supervisor Christos Georgiadis. Thanks to him I had the luck to receive excellent guidance towards the hard path to the PhD degree and being fortunate enough to have Elias Pimenidis and Emmanouil Stiakakis assigned as my committee members. Two persons that have been patient enough to offer me support and guidance not only as supervisors but as true friends.

Many thanks go to the thesis defense committee members for their useful comments and feedback that helped making this thesis of a higher standard. Additionally, I would like to thank the anonymous reviewers of the papers that have been submitted in various journals and conferences, who have provided valuable feedback that improved the quality of this work.

Finally, I would like to express my gratitude towards the staff of the department of applied for providing various types of help all these years.

Νίκος Πολατίδης

Contents

List of Publications

Parts of this thesis and the general work from the presented research has been published in the following journals and conferences:

Journals

J7. Polatidis N., Georgiadis C.K., Pimenidis E., Stiakakis E. 2017. Privacy-preserving recommendations in context-aware mobile environments. *Information and Computer Security*, 25(1), pp. 62-79.

J6. Polatidis N., Georgiadis C.K., Pimenidis E., Mouratidis H. 2017. Privacy-preserving collaborative recommendations based on random perturbations. *Expert Systems with Applications,* vol. 71, pp. 18-25. **(Impact Factor: 2.981)**

J5. Polatidis N. and Georgiadis C.K. 2017. A dynamic multi-level collaborative filtering method for improved recommendations. *Computer Standards & Interfaces,* vol. 51, pp. 14-25. **(Impact Factor: 1.268)**

J4. Georgiadis C.K., Polatidis N., Mouratidis H., Pimenidis E. 2017.A method for privacy-preserving collaborative filtering recommendations. *Journal of Universal Computer Science,* 23(2), pp. 146-166. **(Impact Factor: 0.546)**

J3. Polatidis N. and Georgiadis C.K. 2016. A multi-level collaborative filtering method that improves recommendations. *Expert Systems with Applications*, 48, pp. 100-110. **(Impact Factor: 2.981)**

J2. Polatidis N. and Georgiadis, C.K. 2015. A ubiquitous recommender system based on collaborative filtering and social networking data. *International Journal of Intelligent Engineering Informatics*, 3(2-3), pp.186-206.

J1. Polatidis N. and Georgiadis C.K. 2013. Recommender Systems: The Importance of Personalization in E-Business Environments. *International Journal of E-Entrepreneurship and Innovation*, 4(4), pp.32-46.

Conferences

C3. Polatidis N., Georgiadis, C.K., Pimenidis, E., Stiakakis, E., 2015. A method for privacy-preserving context-aware mobile recommendations. In *6th International Conference on E-Democracy–Citizen Rights in the World of the New Computing Paradigms*. **Springer CCIS** 570 pp. 62-76.

C2. Polatidis N. and Georgiadis C.K. 2014. Factors Influencing the Quality of the User Experience in Ubiquitous Recommender Systems. In *Distributed, Ambient, and Pervasive Interactions* (DAPI 2014). **Springer LNCS** 830 pp. 369-379.

C1. Polatidis N. and Georgiadis C.K. 2013, September. Mobile recommender systems: An overview of technologies and challenges. In *2nd International Conference on Informatics and Applications* (ICIA 2013) pp. 282-287. **IEEE** proceedings.

List of Figures

List of Tables

1 Introduction

Recommender systems research is becoming increasingly important in e-commerce environments. Since their emergence popular recommendation systems exist and include Netflix, Amazon, YouTube, MovieLens and Epinions among others. Furthermore, well-known recommendation libraries include Apache Mahout, Lenskit and Recsys. More specifically, a recommender system uses a certain algorithm or techniques in order to suggest items or services of interest to users, thus aiming to solve in a way the information overload problem found in the World Wide Web (Lu et al. 2015; Jannach et al. 2010; Bobadilla et al. 2013; Polatidis & Georgiadis 2013b; Sivapalan et al. 2014; Georgiadis et al. 2017). The recommendations include movies, books, songs, software, tourism related material, jokes, general products, research papers and even people to people in social networks (Beel et al. 2013; Bobadilla et al. 2013; Lu et al. 2015; Beel et al. 2015; Gavalas et al. 2013; Gavalas et al. 2014). The recommender system adapts to each user in a personalized way to provide recommendations, which is usually done through previous preferences, current interests or a combination of methods. The provided recommendations are suggestions of items or services that could be of potential interest to the user. For example, in a movie recommender system scenario the list of recommended items would include a set of movies that are expected to be of an interest to the user.

Recommender systems, depending on the method they employ can be classified in one of the following categories (J. Bobadilla, Ortega, Hernando, & Gutirrez, 2013; Jannach et al., 2010; Shi, Larson, & Hanjalic, 2014; Su & Khoshgoftaar, 2009):

1. Collaborative filtering: These recommenders suggest items to users that other users with a similar rating history have liked in the past.
2. Content-based: These recommenders suggest items to users that are similar to the items that user has liked in the past.
3. Knowledge-based: These recommenders suggest items to users based either on inferences about the preferences of users or by utilizing specific domain knowledge.
4. Hybrid: These recommenders are based on a combination of two or more algorithms.

However, in mobile recommendation scenarios apart from the recommendation method employed other important aspects include the utilization of context variables and the protection of user privacy (Rodríguez-Hernández & Ilarri 2016; Gavalas et al. 2014; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Liu et al. 2013; Polatidis & Georgiadis 2013a). Thus, if a recommended item is

relevant this is due to the recommendation method and the context. Therefore, mobile recommender systems should make suggestions to users based on a recommendation method, the available contextual information and in addition user privacy needs to be preserved. An overview of the factors affecting mobile recommender systems can be found in section 2.

1.1 Research questions

The general aim of the thesis is to provide a set of components that can be used for developing mobile recommender systems. The author considers that the improvement of fundamental parts of the recommendation process in mobile environments is the most important aspect. To do this, the following research questions have been answered:

RQ1: How can the quality of collaborative filtering recommendations improved?

RQ2: How can user privacy be protected when collaborative filtering systems are used?

RQ3: How can user privacy be protected when context parameters are utilized?

1.1.1 Research aim and objectives

The aim of this research is to propose and develop a model for mobile recommender systems that delivers high quality recommendations, while user privacy is protected.

The research objectives are:

- Analyze the factors that affect the recommendations in mobile scenarios.
- Identify the factors that contribute to the development of mobile recommender systems.
- Employ these factors to propose and validate a set of components that contribute to the development of mobile recommender systems.

In the background section the analysis of previous work shows that mobile recommender systems are developed with a specific recommendation scenario in mind and that all share common characteristics, while a clear understanding of these is missing from the literature. Additionally, relevant work shows that collaborative filtering is the most widely used recommendation method. Thus, a part of the research is concentrated on advancing collaborative filtering by proposing the use of multiple levels within the method. Two different multi-level recommendation methods are proposed. The first one is based on a static model whereas the second one is based on a dynamic approach. However, research has shown that privacy concerns have been identified within users of recommender systems. In more particular, there are concerns about the privacy of user submitted ratings and concerns regarding privacy when context variables are utilized in mobile

recommendation scenarios. Hence, two core parts of the research concentrate on protecting user privacy. On the one hand, the concept of multi-level rating privacy for collaborative filtering systems is introduced and on the other hand the concept of realistic dummy creation for context-aware privacy is introduced.

1.2 Contributions

Mobile recommender systems is still an active area of research and while a mobile recommendation research domain does not exist, previous research is based on modified e-commerce recommendation techniques for specific mobile recommendation environments (Lathia 2015; Jannach et al. 2010; Polatidis & Georgiadis 2015). Furthermore, to support the approach that a mobile recommendation research domain does not yet exist the author developed a ubiquitous recommender system based on collaborative filtering and social networking data, the details of which can be found in the background section. Thus, the author concentrates on how to assist in the development of high quality mobile recommender systems.

This work concentrates on solving the following scientific problem:

How can recommendations of high quality be provided, while user privacy is protected?

The author assists in the development of mobile recommender systems by providing:

1. A static multi-level collaborative filtering recommendation method found in **Chapter 3**.

2. A dynamic multi-level collaborative filtering recommendation method found in **Chapter 4**.

3. A method for rating privacy that is based on random perturbations found in **Chapter 5**.

4. A method for preserving privacy in context-aware mobile scenarios found in **Chapter 6**.

1.3 Structure

The thesis is structured as follows:

- Chapter 1 introduces the thesis by presenting the motivation, research goals and contributions.
- Chapter 2 provides a background on mobile recommender systems. Firstly, the author provides a general overview of the state-of-the art in mobile recommender systems. Secondly, the author presents and analyzes the factors that affect the recommendations. Finally, the author presents an example motivating scenario and a ubiquitous recommender system.
- Chapter 3 proposes a static multi-level collaborative filtering recommendation method.
- Chapter 4 proposes a dynamic multi-level collaborative filtering recommendation method.
- Chapter 5 proposes a multi-level privacy preservation method for collaborative filtering.
- Chapter 6 proposes a privacy protection method for context-aware mobile recommendations.
- Chapter 7 presents the discussion.
- Chapter 8 contains the conclusions.

2 Background

2.1 Literature review

Recommender systems are usually based on a 2-dimensional setting to provide recommendations to users. This means that typical contextual information such as the time, the weather and the location of the user requesting the recommendations are ignored. Furthermore, users mostly use a mobile device such a smartphone or tablet for most of their tasks and perform those while on the move. Mobile recommender systems are based on a recommendation algorithm and contextual information to provide recommendations of items or services to users of mobile devices. Thus, a set of components that can be utilized to facilitate in the process of developing mobile recommender systems is delivered.

Several problems have been identified and solved to a certain extend. For example, in (Lathia 2015) an overview of mobile recommender systems is given with the outcome that a mobile recommendation domain does not exist and that every mobile recommender system is developed with a specific task in mind. This argument is also present in other works in the literature such as (Jannach et al. 2010; Rodríguez-Hernández & Ilarri 2016; Liu et al. 2013). In addition, certain works such as Hybreed (Hussein et al. 2014) develop on this concept. In Hybreed the authors propose a framework for the automatic delivery of context-aware hybrid recommender systems. Also, the term UbiCars has been proposed by (Mettouris & Papadopoulos 2014). In this survey work, the characteristics of ubiquitous computing and recommender systems are described in detail and the authors explain what ubiquitous recommender systems are. Another, relevant work has been proposed by (Rodríguez-Hernández & Ilarri 2016). In this work, the process of developing pull-based mobile recommender systems is explained in detail and a generic mobile recommendation architecture is proposed and evaluated. Furthermore, one of the major issues identified is the absence of good relevant datasets in the literature and that most datasets are domain specific. Consequently, an important problem identified it the lack of a good quality datasets for the purpose need and the research is based on specific mobile domains. These include the development of mobile recommender systems for the recommendations of generic commerce items like books, movies, music and photos or the recommendations of points of interest (POIs) such as restaurants and tourist attractions. Thus, the remaining part of this chapter concentrates on the coverage of related works found both in the general commerce and in the tourism domains respectively.

Mobile recommender systems for the recommendation of books, photos and music have been proposed in the literature. For example, CoMeR (Yu et al. 2006) provides the recommendation of different type of media to its users using a context-aware approach. Also, in (Lemos et al. 2012)

the prototype version of a photo recommender system can be found. It is a mobile recommender system for photos which utilizes current contextual data in combination with information found in the photos. Another interesting work is found in (Baltrunas et al. 2011) with the name of InCarMusic. A context-aware recommender system is used to provide music recommendations to the passengers of a car. In addition, another similar work for the recommendations of music depending on the daily activities of a person is provided by (Wang et al. 2012). This mobile recommender is based on a probabilistic model to propose songs to users of mobile devices depending on their current activity. Another example of a mobile recommender system can be found in the mobile news domain that is based on the current context and the format of the recommended news (Sotsenko et al. 2014). Mobile recommender systems have been used for movie show times. RecomMetz (Colombo-Mendoza et al. 2015) is based on a knowledge based recommender and contextual information to recommender movie show times to users of mobile devices. Another interesting m-commerce recommender system for the recommendation of mobile applications based on collaborative filtering and context has been proposed by (J. Lin et al. 2011). Motivate is yet another mobile recommender for the recommendation of personalized activities for the user who wants to maintain a healthy lifestyle (Y. Lin et al. 2011). Furthermore, in mobile recommender different data sources can be utilized to provide personalized recommendations to users. These include the approach proposed by (H. Zhu et al. 2014) were mobile logs are used and the method proposed by (Liu & Liou 2011) that utilizes data from multiple mobile channels. Thus, it is noticeable that different recommendation methods and data sources can be used along with contextual information for servicing users.

SMARTMUSEUM is a mobile recommender system proposed by (Ruotsalo et al. 2013). The aim of this recommender is to provide recommendations for heritage related items in indoor and outdoor scenarios. To achieve this, different type of contextual information is utilized such as the location and time. Another mobile recommender system based on collaborative filtering and context data is iTravel (Yang & Hwang 2013). This is a peer-to-peer recommender for the recommendation of travel attractions. On the other hand, Turist is a mobile recommender systems for recommending cultural and leisure activities once the user is at the destination (Batet et al. 2012). Additionally, mobile recommender systems can be used to provide pushed recommendations without explicit requests from the user. The recommender recommends item or services when the context is appropriate for those. Such a recommender system has been proposed by (Woerndl et al. 2011). Furthermore, context-aware collaborative filtering has been used by (Huang & Gartner 2012) for the recommendation of mobile guides. In this work the authors recommend similar POIs to users with similar interests in similar contexts. A mobile recommender system based on banking data history has been proposed (Gallego & Huecas 2012). The mobile

application recommender places to users that have visited before and paid with their credit card. A modified collaborative filtering approach that uses several contextual variables has been proposed to provide recommendations using a decision tree and a set of rules (Kim et al. 2010). A mobile recommender system for guides that is based on collaborative filtering and contextual information has been proposed by (Gavalas & Kenteris 2011).

2.2 Factors affecting mobile recommender systems

Three influencing factors exist that can affect mobile recommender systems and their ability to provide accurate personalized recommendations. These include the recommendation method, the context and privacy concerns (Polatidis & Georgiadis 2014). In addition, there are additional but less influencing factors that have been identified and explained. The author concentrates on improving aspects of the influencing factors.

2.2.1 Influencing factors

Recommendation method. Recommender systems rely in some form of recommendation method to suggest the appropriate products or services to the user. The most important recommendation methods include (Bobadilla et al. 2013; Lu et al. 2015): Collaborative filtering which is a method that recommends items to users that other users with similar ratings have liked them in the past. This works by asking each user to submit ratings for products or services and then searches between the ratings for similar users and provides the recommendations (Shi et al. 2014). Content-based filtering which is a method that uses a set of keywords supplied by the user that can be matched in the item's description (Konstan & Riedl 2012; Bobadilla et al. 2013). Finally, hybrid is a method that uses a combination of two or more recommendations methods (Jannach et al. 2010; Bobadilla et al. 2013).

Context. Context variables are utilized by mobile recommender systems to provide more accurate and personalized recommendations. It is a type of data that is necessary to users that move constantly and their status changes. Different types of context can be employed in mobile scenarios and include, among others, location, time, weather and social presence. Contextual information is important for location-based recommendations (Adomavicius & Tuzhilin 2015; Ricci 2010). Information can be collected either explicitly, by asking the user to provide data, or implicitly by collecting data from the mobile device and related sensors, such as the Global Positioning System.

The context can be applied by using three different ways (Adomavicius & Tuzhilin 2015): First, Contextual pre-filtering is a method where the contextual data is used to filter out irrelevant data from the dataset and then apply the recommendation method. Also, Contextual post-filtering is a method where the recommendation method takes places and then the irrelevant data are filtered out. Finally, Contextual modeling is a method where the recommendation method is designed in a way that the context is utilized within.

Privacy. Mobile recommender systems offer the benefit of providing personalized recommendations to users of a context that constantly changes. On the other hand, the ways that user data might be processed direct users towards a negative attitude, when it comes to supplying personal contextual information (Mettouris & Papadopoulos 2014). Privacy protection techniques have relied mainly on location-based services and do not take into consideration the whole concept of context (Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Polatidis et al. 2015). Privacy is an important factor that can be addressed properly using the right methods and makes it possible for the user to supply the required contextual information, thus making both the system usable and receive highly accurate personalized recommendations.

2.2.2 Less influencing factors

Perceived accuracy. A factor that needs some consideration is perceived accuracy which is a point where a user feels that the recommendations match his preferences (Pu & Chen 2011). It is a measuring assessment of how good the recommender performed and how accurate is to find the interests of a user.

Familiarity and novelty. Familiarity is a description of the previous experience that user had with the recommended item or service (Pu & Chen 2011). However, familiarity might mean that all the recommendation categories must be familiar to the user. Novelty must be introduced and balanced with familiarity so the user would be as satisfied as possible.

Attractiveness. Attractiveness is conserved with the process of irritating the user and evoke positive imaginations and increase the possibility of desiring. Attractiveness is concerned on how well the recommendations will be delivered to the user and not the recommendations provided (Pu & Chen 2011).

User interface. Limitations found in the user interface, where different devices may be used, the task would be to develop suitable and user friendly interfaces (Gavalas et al. 2014). User interfaces are tightly related to the attractiveness as described above and could improve the quality. The more attractive is the user interface the user will be satisfied more.

New user and new item. The new user and item problem are very important when the algorithm used is based solely in collaborative filtering (CF). They occur when a new user or item is added to the database there is no history about the user or no rating history about the item or service. If a user wants higher quality from a recommender from the very beginning of joining a service, then this is a very important issue that needs to be faced and this can be dealt with the use of hybrid algorithm that utilize data from social networks (Massa & Avesani 2007).

Multilingual personalization. Given the fact that there is a vast amount of data found on the internet, these data can exist in different languages (Ghorab et al. 2011). It is possible that the data requested from a user will not be available in his native language but be available in a foreign language. Research has been done towards the field of personalized multilingual information retrieval (Ghorab et al. 2011). It is a field where if suitable research occurs then more useful recommendations could be delivered.

2.3 Motivating scenario

This section describes a motivating scenario that illustrates the necessity for a privacy-preserving context-aware mobile recommendation architecture. The scenario shows the main benefits that a user can gain if a mobile recommender system is used and that a privacy-preserving system is necessary to assist the user in the process of gaining those benefits. The author follows an example from a fictional user to describe the motivating scenario which assumes the existence of a mobile application, MobiRec (Mobile Recommender), which can be installed in mobile devices, such as smartphones or Tablets. This application recommends movies of interests to the user, considering past common ratings between users and available context parameters.

2.3.1 Example scenario

Bob is at home at 7:30pm. It is Saturday and the weather is rainy. Bob is relaxing with some of his friends while they are deciding to watch a movie. Bob then chooses to use MobiRec to assist him with finding a relevant movie to watch with his friends. He opens the applications and selects the option of receiving recommendations of movies to his screen. Bob want to get the best possible

recommendations while his privacy is respected. Then, automatically, MobiRec enriches the input with available contextual information such as the hour of day, the location, company and weather. The mobile recommender then communicates with the central database where the ratings of users about movies are stored, passes the contextual information to the server so the most relevant context-aware recommendations are provided. At this moment, the server is aware about private user information, which somehow need to be protected from unauthorized use. Thus, MobiRec uses the method described in section 5 to create a set of dummy parameters that are passed to the server among the real parameters. MobiRec also has the option for extra privacy under its settings by using the privacy-aware interface, which in this case utilizes the context to see if the user is alone or with company to provide a warning saying that other people might have a look at the recommendations and if with the press of a button the list of the recommendations is released to the screen.

Then, at the end of the movie Bob is asked by MobiRec to rate the movie that he just watched, so better recommendations can be provided in the future. The appropriate user interface pop-ups in the screen where Bob can select a numerical value. However, if the extra privacy is selected under the parameters of the mobile recommender application then a different rating submission policy applies. Privacy can be threatened from nearby people staring at the screen of the mobile device. To avoid any breach of privacy the entered value of the ratings is manipulated with the method described in section 5. Now, Bob can submit a rating freely with him only knowing the real value passed to the server and while his friends are watching.

2.4 A ubiquitous recommender system

The author developed a ubiquitous recommender system to support the fact that a mobile recommendation domain does not exist and that mobile recommender systems are developed with a specific scenario in mind. The ubiquitous recommender system is based on collaborative filtering and social networking data derived from the Epinions website (Massa & Avesani 2007). It is further explained, analyzed and evaluated in depth in (Polatidis & Georgiadis 2015).

User experience becoming more and more an essential part in the attention of the research community. However, there isn't much work done on how the quality of the user experience in ubiquitous recommender systems can be increased and what kind of standards could be specified to work towards that direction. The criteria need to be combined into a comprehensive framework that could be potentially used to provide better quality ubiquitous recommender systems. The framework should take into consideration all the major criteria which should be satisfied. A

comprehensive model identifying all the essential qualities could be established as a standard, which will convince potential users to adapt such a system.

2.4.1 Problem statement

Nowadays with the growth of the internet and the development of high capability mobile devices the information overload problem is becoming serious. Recommender systems have become widely known and used in recent years to overcome this problem, with the use of collaborative filtering as the most widely known and used (Ekstrand et al. 2011). Furthermore, the technology nowadays has become ubiquitous and a clear majority of users tend to use a mobile device to use the internet. All these users need recommendation technologies that can be used in their device by taking in consideration a broader context. However there exist several important factors that influence the quality of the user experience and should be handled. Most notable issues can be found in collaborative filtering it is not capable of making any predictions about new users, which have not rated any products or services yet and about new items that have not received any ratings yet. To address this problem the use of data from a social rating network is considered in combination with the collaborative filtering method. Social rating networks are made of a rating network and a friendship network, which means that still recommendations can be made even when there are no ratings available. Furthermore, privacy is an issue that is important to users of ubiquitous recommender systems. Finally, is the use of contextual parameters since the main usage target will be recommendations in ubiquitous environments.

2.4.2 Collaborative filtering preliminaries

Collaborative filtering is the most widely used approach in recommender systems (Shi et al. 2014; Polatidis & Georgiadis 2017). The recommender using this approach will make recommendations based on users that have similar preferences or tastes using ratings provided from those users. The overall idea is to make recommendations that the user is likely to be interested. In user based collaborative filtering a database is created, which contains the nearest neighbor of the user requesting the recommendations. It is a simple idea where a table stores the user id, the item id and the rating. Table 2.1 is such an example. Then the algorithm will identify similar users using a similarity function. Equation 2.1 represents the Pearson correlation. Sim (a, b) is the similarity of users a and b, $r_{a,p}$ is the rating of user a for product p, $r_{b,p}$ is the rating of user b for product p and $\bar{r}a, \bar{r}b$ represent user's average ratings. P is the set of all products.

	Product1	Product2	Product3	Product4	Product5
User1	5	3	4	4	Empty
User2	3	1	2	3	3
User3	4	3	4	3	5
User4	3	3	1	5	4
User5	1	5	5	2	1

Table 2-1 Product ratings

$$Sim\ {}^{PCC}_{a,b} = \frac{\sum p \in P(ra, p - \bar{r}a)(rb, p - \bar{r}b)}{\sqrt{\sum p \in P(ra, p - \bar{r}a)^2}\sqrt{\sum p \in P(rb, p - \bar{r}b)^2}} \quad (2.1)$$

If we want to calculate similarities of users to user number 1. Then the similarity values are created, ranging from -1 to 1. As shown in table 2.2 the user closest to User1 is User3.

	User1	User2	User3	User4	User5
User1	1	0.70	0.85	0.2	-0.79

Table 2-2 similarity table

2.4.3 Social rating network preliminaries

A social rating network is a service that helps people to connect between them, exchange information and most importantly rate products (Massa & Avesani 2007). One of the most know social rating networks is Epinions. In such a network, a Truster-Trustee network is created and User1 trusts User2. Although this does not mean that User2 trusts User1. It is a one-way network. See table 3 for such a network example.

Truster	Trustee
1	2
2	3
3	2

Table 2-3 Truster-Trustee network

Then a different number of methods can be applied to this data and include get recommendations. From the trust network only could be an option or go further down to the network and include

friend of a friend. Finally, similarity methods can be applied that perform different techniques to retrieve the nearest user neighborhood. Finally, it should be highlighted that two different types of social relationships exist in social rating network. The first one is the user, item and rating relationship and has been described in section 4.2 and the second one is the Truster-trustee network.

2.4.4 Proposed method

It should be noted that the quality of recommendations and hence an increased user experience is heavily based on the algorithm used. A hybrid algorithm based on collaborative filtering is necessary due to the better prediction of such algorithms. However, there is a problem in collaborative filtering with the new user and item issues, which can be solved with the use of data from social rating networks such as Epinions. In addition, the proposed algorithm will incorporate contextual information that is useful in ubiquitous environments. Figure 2.1 gives a high-level architecture of the proposed recommender system that utilizes social media data.

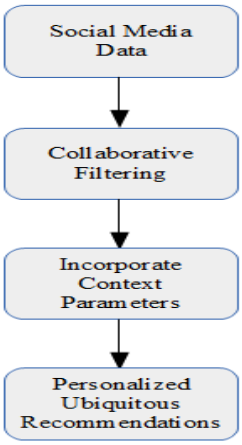


Figure 2-1 Proposed Ubiquitous Recommendation Architecture

The author uses the data from the social rating network, which includes both the user-item network and the user trust network. The next step is to use the context parameters desired by the user, which is also known as contextual post filtering. Ubiquitous recommenders need to be context aware to be effective and capable of providing the correct results. Context as defined by (Adomavicius & Tuzhilin 2015) includes parameters such as:

- Location
- Time
- Date
- Weather
- Any other useful information

The contextual information is very important in ubiquitous environments and is crucial for location based services. (Adomavicius & Tuzhilin 2015) states that context can be incorporated either:

- Explicitly from the user
- Implicitly from changes in the environment such as location change
- Using data mining or statistical methods

The first part of the proposed model is to utilize the data from the user-item rating network to identify the k-nearest neighbors of the user who is requesting the recommendations. This is done using equation 1 with a pre-defined number of user neighbors. The next step is to use the information acquired from the user-trust network in order to incorporate the information from the network (Who the user trusts) into the rating network. The values a, b are users, UTA it the set of the trust network of user a. UR is the set of all users and ratings. Equation 2.2 describes the definition of the enchased trust-based similarity.

$$Sim\ (a,b) = \begin{cases} Similarity\ value\ using\ equation\ 1, & if\ b \notin\ UTA \\ Similarity\ value\ using\ equation\ 1 + 0.50, & if\ b\ \in UTA \\ 0, & otherwise \end{cases} \quad (2.2)$$

The algorithm modifies in positive manner the similarity value returned. If the comparing user belongs to the trust network of user requesting the recommendations then the similarity a value of 0.50 is added to the similarity value. It is noted that after experimentation using values from 0.1 to 0.6 the value 0.5 returned the best result for the Epinions dataset, which makes the accuracy of the algorithm better when compared to classical collaborative filtering. Further details are shown in the evaluation section. Moreover, it should also be noted that if the addition returns a value greater than 1 then it is automatically converted to 1, which is the maximum. Also values above 0.60 where not used due to the reason that the method would only recommend items based on the trust network of the user. Algorithm 2.1 provides the pseudocode.

Algorithm 2.1 Combining rating and trust network for user a ∈ U

Input: UR → the set of all users and ratings
UTA → the set of the user-trust network of user a
Output: final similarity

```
    for the size of UR
    Sim (a, i) // the similarity function using equation 2.1
    tempSimilarity = Sim (a, i) // a value between -1 to 1
        if (i.isIn (UTA))
        tempSimilarity + 0.50
        finalSimilarity = tempSimilarity
        else finalSimilarity=tempSimilarity
    end for
return finalSimilarity
```

At the next step, after the similarity values are computed, a list of products is generated and context parameters are applied to sort out irrelevant information.

2.5 Synopsis

By taking into consideration the previous related works and the ubiquitous recommender system developed by the author it is shown that classical web recommendation technologies are combined according to a scenario and a set of derived requirements from the scenario are applied to achieve a specific purpose in a specific environment. More specifically, the developer can choose to utilize any recommendation component derived from web-based recommender systems to fulfill the requirements. Consequently, and as it has been explained previously, the author analyzed the influencing and less influencing factors that affect mobile recommender systems and delivers a set of novel components that improve each of the influencing factors. Thus, each component targets each one of the influencing factors and it is shown that each of the components improve the process at the specific recommendation process part. Moreover, the proposed components could be used separately or integrated when developing a mobile recommender system to improve the overall quality of the process in terms of quality and privacy.

3 Multi-level collaborative filtering

Collaborative filtering is one of the most used approaches for providing recommendations in various online environments. Even though collaborative recommendation methods have been widely utilized due to their simplicity and ease of use, accuracy is still an issue. In this chapter, the author proposes a multi-level recommendation method with its main purpose being to assist users in decision making by providing recommendations of better quality. The proposed method can be applied in different online domains that use collaborative recommender systems, thus improving the overall user experience. The efficiency of the proposed method is shown by providing an extensive experimental evaluation and with comparisons to alternatives.

3.1 Introduction

Nowadays, more and more people find that the constant growth of the web in combination with the development of technologies such as smartphones and Tablets results in spending more time accessing information online. However, these developments have brought a massive amount of information, resulting in an information overload problem (Moradi & Ahmadian 2015; Bobadilla et al. 2013). With too much information all over the web, users find it very challenging to find the data they need. For that reason, most users find it frustrating when looking online for what movie to watch, who to add as a friend in a social network and many other related search problems. The solution to this problem is recommender systems, which apply techniques developed to analyze user data and make recommendations that the user will probably like. It is a method that both the service provider and the user are benefited from. The main reasons for the mutual benefits include the fast processing of data for the service provider, the higher percentage of sales, the saving of time for the user and the discovery of products or services that otherwise would be difficult to find (Polatidis & Georgiadis, 2013).

Collaborative filtering is the most known and widely used technique for providing fast and accurate enough recommendations to users (Shi et al. 2014; Ekstrand et al. 2011; Konstan & Riedl 2012; Su & Khoshgoftaar 2009). This method relies on a database of ratings submitted by each user for products or services, then the ratings are compared to each other with the use of suitable similarity method to provide recommendations to the user who makes the request. The main two functions of such systems are to identify a pre-specified number of neighbors according to similar ratings and then provide the recommendations. Collaborative filtering has been widely adopted by many real-world systems, such as Netflix and Amazon (Wang et al. 2015), and this is due to its simplicity and efficiency.

In addition to collaborative filtering, other recommendations methods include content-based, which is based on item metadata. In this method the user supplies a set of information and preferences and the algorithm makes recommendations according to the settings provided (Bobadilla et al. 2013) and (Burke 2007). Moreover, knowledge-based recommender is another recommendation approach that uses inferences about user preferences and specific knowledge about the domain and also how the items or services to be recommended meet the preferences set by the users (Jannach et al. 2010). A widely used recommendation approach is the combination of one or more recommendation methods, and is called hybrid (Burke 2002). It doesn't necessarily mean that the two methods must be different, but they could be two different collaborative filtering methods as well (Jannach et al. 2010; Burke 2007).

In most collaborative recommendation systems a similarity function, such as Pearson Correlation Coefficient (PCC) or Cosine (Shi et al. 2014), is utilized by the system to provide recommendations by taking in consideration the absolute ratings between users. The motivation is to divide user similarity, as offered by PCC, into different levels and add constraints to each level. Is it shown that by modifying the user similarity, which is a value from -1 to 1 according to the constraints that each user belongs, the accuracy of the recommendations is improved. Furthermore, the author argues that the quality of the recommendations is improved as well when the constraints are at place. The proposed method attempts to provide recommendations of better accuracy and quality when compared to other alternatives. However, for this to be done correctly, enough ratings should have already been submitted by users of the system.

3.2 Related work

While collaborative filtering methods have been widely used by many real-world systems, including Netflix and Amazon, there are not sufficient details available regarding the provided recommendations. Collaborative filtering techniques use a database of ratings among users and items, such as the one shown in Table 3.1, must be present (Shi et al. 2014).

	Item/Service 1	Item/Service 2	Item/Service 3	Item/Service 4	Item/Service 5
User 1	-	2	3	5	-
User 2	1	2	4	5	5
User 3	2	5	1	1	2
User 4	3	-	-	-	3
User 5	-	3	4	-	-

Table 3-1 A database of ratings

When recommendations need to be generated for a user, then the ratings are loaded into memory and a similarity function is used. The main part is how to estimate the similarity value between two users. This is called neighborhood identification and the job of the similarity function is to firstly identify a pre-specified of *k* nearest neighbors based on their similarity value. In present recommendations systems, the value of *k* can vary from a few, possibly 2 to 5, to as many as possible with the number ranging from 10 to 20 to 30 and so on up to hundreds of neighbors. A high number of neighbors doesn't necessarily mean that the accuracy of the recommendations will be high though.

Now, as mentioned for the identification of the nearest neighbors a similarity function such as PCC is necessary to be used. PCC is defined in Eq. 3.1. In PCC, the sum of ratings between two users is compared. Sim (a, b) is the similarity between users a and b, also $r_{a,p}$ is the rating of user a for product p, $r_{b,p}$ is the rating of user b for product p and $\bar{r}a$ and $\bar{r}b$ represent the user's average ratings. P is the set of all products. Moreover, the similarity value ranges from -1 to 1 and higher is better.

$$Sim\,{}^{PCC}_{a,b} = \frac{\sum p \in P(ra,p - \bar{r}a)(rb,p - \bar{r}b)}{\sqrt{\sum p \in P(ra,p - \bar{r}a)^2}\sqrt{\sum p \in P(rb,p - \bar{r}b)^2}} \quad (3.1)$$

After the similarity values are computed using Eq. 3.1 used and the formation of the *k* nearest neighborhood, then the rating procedure takes place, where ratings are being predicted for items. The items with the highest rating predicted value are being recommended to the user who made the request. On the other hand, PCC has inspired other similarity methods such as the weighted PCC (WPCC) (Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl 1999) defined in Eq. 3.2. WPCC is based on PCC and provides recommendations based additionally on the number *T* of co-

rated items between users. In their work, the value was set to 50, which means that if the number of the co-rated items was 50 or more the recommendations from these users are preferred. Also in the case that less ratings are available then the algorithm switches in ordinary PCC.

$$Sim_{a,b}^{WPCC} = \begin{cases} \frac{|Ia \cap Ib|}{T} \cdot Sim_{a,b}^{PCC}, & if \left|Ia \bigcap Ib\right| < T \\ Sim_{a,b}^{PCC}, & otherwise \end{cases} \quad (3.2)$$

A somewhat similar approach for identifying neighbors is proposed by (Jamali & Ester 2009) and is defined in Eq. 3.3. In this method, the similarity of small number of co-rated items is weaken.

$$Sim_{a,b}^{SPCC} = \frac{1}{1 + \exp(-|Ia \cap Ib|/2)} \cdot Sim_{a,b}^{PCC} \quad (3.3)$$

Another approach to recommendations based on collaborative filtering is Jaccard similarity (Koutrika et al. 2009). In this approach, the similarity computation of PCC is not used, but only the number of co-rated items is taken into consideration. Jaccard similarity is defined in Eq. 3.4.

$$Sim_{a,b}^{Jaccard} = \frac{|Ia \cap Ib|}{|Ia \cup Ib|} \quad (3.4)$$

Other proposed similarities for collaborative filtering include the mean squared difference (MSD) (Cacheda et al. 2011). This method captures the difference that the users have in their ratings. Furthermore, another proposed similarity measure has been proposed by (Lu et al. 2013) where the use of fuzzy set theory is used with the aim to assign different weight values to different rating differences. Another method, proposed by (Wang et al. 2015), uses entropy to provide user similarity in collaborative filtering. In this work the majority of ratings used by PCC must be similar. (Liu et al. 2014) proposed a collaborative filtering improvement that does not only consider the local user rating information, but also the global behavior of the user. (Son 2014) proposed a fuzzy recommendation method that uses demographic data instead of user ratings. One more similarity measure found in the literature is proposed by (Bobadilla, Ortega, Hernando & Bernal 2012). This recommendation method tries to solve the cold start problems found in recommender systems. It aims to solve the problem of new users stop using the system because of low accuracy found at the initial stages (when just a few ratings are available). For this reason, the authors

propose a new similarity measure based on neural learning and uses optimization that provides better accuracy to users with few submitted ratings. An alternative similarity measure based on singularities is described in (Bobadilla, Ortega & Hernando 2012). In this approach, contextual information derived from users is used to calculate the singularity for each item. According to their results the similarity is improved when compared to simple collaborative filtering. (Ahn 2008) offers, yet, another similarity measure that alleviates the new user cold start problem. This paper proposes a heuristic measure that improves the similarity under cold start scenarios, and in particular when only few ratings are available. (Anand & Bharadwaj 2011) proposed a recommendation method that utilizes sparsity measures based on both local and global similarities and it is still another method that improves the accuracy of recommendations that are based on collaborative filtering. Moreover, recommendation methods that aim to improve the accuracy of collaborative recommendations but the approach offered by (Moradi & Ahmadian 2015) is different, since it uses data from both ratings submitted from users and from social-trust networks that are part of the recommendation system. Also, another approach that uses not only trust but distrust information as well, is the one proposed by (Fang et al. 2015). Moreover, (Polatidis, N., & Georgiadis C.K, 2015) offer an improved collaborative filtering similarity that is based on data from social rating networks and also show how this method can be used in ubiquitous environments and also preserve the user privacy if necessary. (Toledo, Mota, & Martinez, 2015) proposed a recommendation method that aims to detect and correct unreliable ratings that could bias the recommendations provided and to do it the method characterizes the users from their profiles. Finally, it should be noted that Cosine similarity is used also widely by collaborative recommender systems and its main difference with PCC is that it does not take into consideration user habits with extremely low or high ratings (Shi et al. 2014; Ekstrand et al. 2011).

3.3 Proposed method

In this section, the author gives a description of the proposed method. The first step is to introduce the multi-level function, that aims to improve the accuracy of the recommendations, and then to address the problem where the function is not able to provide recommendations for certain users (Polatidis & Georgiadis 2016).

3.3.1 The steps of the proposed method

PCC is the most widely used and accurate similarity measure used by collaborative filtering recommender systems (Wang et al. 2015). The goal is the enhancement of the PCC similarity measure, which results in the improvement of the accuracy performance. The difference among the ratings of the items that users have co-rated is the value of similarity that exists between these

users. Many similarity measures get better accuracy by manipulating how these calculations take place and the methods are based on absolute rating differences. In this approach, it is crucial that all of the values of co-rated items between two users must be the same. See Table 3.2 for an example where it is shown that 'user 2 new' has a different value for item 5. This means that there is no similarity between 'user 2 new' and 'user 1' (while 'user 2' is similar with 'user 1').

	User 1	User 2	User 2 new
Item 1	5	5	5
Item 2	4	4	4
Item 3	3	3	3
Item 4	2	2	2
Item 5	1	1	5

Table 3-2 Absolute ratings

While other methods use the method of absolute ratings either by adding weights, or by manipulating different variables to achieve a better result, the approach is based on a multi-level division. Furthermore, the author argues that every user has a rating expression that should not be punished that way. Also, it is agreed that with PCC based similarity approaches that consider higher the similarity when the values of the co-rated items are as close as possible. The proposed method heavily relies on PCC, the similarity value returned from it and the number of co-rated items. To achieve this, initially a similarity function which is defined in Eq. 3.5 is introduced. In this Eq., T stands for the total number of co-rated items, x is a positive real number such as $x \in R$, y is a positive real number such as $y \in R$. In the proposed method, the author argues that the by dividing the algorithm in multiple levels the accuracy of the recommendations is improved. To show the effectiveness of the method, experiments using four levels have been used. In this context, at the first four steps the number of co-rated items is checked and if it is more than the pre-specified thresholds (*t1, t2, t3, t4)*, then it can proceed to the next step to check the similarity value derived for the two users from PCC. For users that enough co-rated items exist and the PCC similarity value is greater than a pre-specified threshold (y) then a list of recommendations is returned. Otherwise, for users that there is not available enough co-rated items, zero value is returned. Finally, *t1, t2, t3* and *t4* are natural numbers that represent the constraints put on the number of co-rated items for each level ($t1 \in N, t2 \in N, t3 \in N, t4 \in N \wedge t4>t3>t2>t1$)

$$Sim_{a,b}^{Proposed} = \begin{cases} Sim_{a,b}^{PCC} + x, & if\ \frac{|Ia \cap Ib|}{T} \geq t1\ and\ Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x, & if\ \frac{|Ia \cap Ib|}{T} < t1\ and\ \frac{|Ia \cap Ib|}{T} \geq t2\ and\ Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x, & if\ \frac{|Ia \cap Ib|}{T} < t2\ and\ \frac{|Ia \cap Ib|}{T} \geq t3\ and\ Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} + x, & if\ \frac{|Ia \cap Ib|}{T} < t3\ and\ \frac{|Ia \cap Ib|}{T} \geq t4\ and\ Sim_{a,b}^{PCC} \geq y \\ 0, & otherwise \end{cases} \quad (3.5)$$

3.3.2 Hybrid approach

The proposed method although it improves the accuracy of recommendations, is unable to provide recommendations to other users that do not have at least a few co-rated items and a certain PCC similarity value. For this reason, a hybrid approach that can switch to PCC only if enough ratings are not available for the multi-level approach to provide recommendations. Algorithm 3.1 provides the hybrid approach.

Algorithm 3.1: A hybrid approach to recommendations

Input: User id /* the id of the user requesting the recommendations */
t /* t is the minimum pre-specified number of common ratings – Valid values are t1, t2, t3 and t4 */
y /* y is the minimum pre-specified similarity value derived from PCC */

Output: Recommendation method

```
Load k nearest neighbors for User id
For (int i=0; i<k.size; i++)
    If k.get(i, n) >= t && s >=y  /* n is the number of co-rated items */
                                  /* s is the similarity value derived from PCC */
        Then Load function defined in Eq. 3.5
    Else
        Load PCC
    End If
End For
```

3.4 Experimental evaluation

In this section, the experimental evaluation of the proposed method takes place and the results are based on five real datasets and widely used metrics with different parameters. The evaluation took place on an Intel i3 2.13 GHz, 4GBs of RAM, running windows 8.1 and all the algorithms were implemented in the Java programming language.

3.4.1 Datasets

Experiments on three real datasets have been conducted to observe the results under different amounts of ratings and users. The datasets are MovieLens (Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl 1999), MovieTweetings (Dooms, S., De Pessemier, T., & Martens 2013) and Epinions (Massa & Avesani 2007).

- **MovieLens.** MovieLens 1m is a real dataset which contains 4000 movies, 6000 users and 1,000,000 ratings. The data have been collected by the University of Minnesota and are associated with their online movie recommendation system. This dataset is one of the many that is publicly available from the University and has been widely used before for offline experimental evaluation of collaborative filtering recommender system performance. The data in the dataset are in the form [userid] [itemid] [rating]. All the rating values are in the scale 1 to 5.
- **MovieTweetings.** This is a publicly available dataset crawled from Twitter. It is a dataset consisted of movie ratings in the scale 0 to 10. The data in the dataset are in the form [userid] [itemid] [rating]. The dataset has 431,780 ratings from 39,363 users on 22,610 items.
- **Epinions.** This is a publicly available dataset crawled from Epinions.com. It is a general product recommendation dataset. The data in the dataset are in the form [userid] [itemid] [rating]. All the rating values are in the scale 1 to 5. The dataset has 664,824 ratings from 49,290 users on 139,738 items.

3.4.2 Comparisons

The following collaborative filtering recommendation methods have been used in the comparisons.

- **PCC.** This is a method that calculates the statistical correlation between the common ratings of two users to determine the similarity between them. The output will be between -1, which is the lowest, and 1, which is the highest possible value. This method is defined in Eq. 3.1.
- **WPCC.** This is a method that is based on PCC. The difference is that the algorithm considers a pre-defined number of common ratings between users. Moreover, if the number of common ratings is not sufficient then it switches in classical PCC.
- **SPCC.** This is a method that is based on PCC. The difference is that the similarity value returned for users with a small number of common items is weaker. However, the function is adjusted to produce a higher similarity to users with a smaller number of co-rated items.

The methods that have been compared against the proposed method have different characteristics. The proposed method is based on multiple levels, from top-to-bottom, with each of these levels having constraints. The constraints provide a higher similarity value between users that have more common items and a PCC similarity value above a certain threshold, which is something that is not available in the other methods. However, it should be noted that if any of the constraints of the level is not satisfied then the similarity value between the users will be set to zero. Thus, the main weakness of the proposed algorithm is that in the case that enough ratings are not available, then none of the levels can be constructed and recommendations cannot be provided. In this case another recommendation algorithm, such as PCC, needs to be applied. Furthermore, Table 3.3 provides a comparison between the methods.

Method	Characteristics
PCC	Statistical correlation between two users providing a similarity value from -1 to 1
WPCC	Based on PCC, gives more weight to users with a pre-specified number of common ratings
SPCC	Extends PCC by providing a weaker similarity value between users depending on co-rated items
Proposed	Enhances the similarity value of users that belong to certain categories and ignores the rest

Table 3-3 Comparison between methods

3.4.3 Measures

For the purpose of measuring the accuracy of the recommendations provided by collaborative filtering recommendations the widely accepted by the research community Mean Absolute Error (MAE) metric has been used (Herlocker et al. 2004; Shani & Gunawardana 2011). MAE is defined in Eq. 3.6, where *pi* is the predicted rating and *ri* is the actual rating in the summation. This method is used for the computation of the deviation between the predicted ratings and the actual ratings. It should also be noted that lower values are better.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|pi - ri| \quad (3.6)$$

In information retrieval systems, such as recommender systems, there are metrics that can measure the quality of the top N recommendations (Herlocker et al. 2004; Shani & Gunawardana 2011; Schröder et al. 2011; A. Bellogin et al. 2011). These metrics are Precision and Recall. For these

metrics, higher values are better. Eq. 3.7 defines the Precision metric, which is the amount of relevant recommendations found in the retrieved set of recommendations and Recall is the amount of relevant recommendations that have been retrieved successfully. Eq. 3.8 contains the definition of Recall metric.

$$precision = \frac{Correctly\ recommended\ items}{Total\ recommender\ items} \quad (3.7)$$

$$recall = \frac{Correctly\ recommended\ items}{Relevant\ items} \quad (3.8)$$

3.4.4 Settings

For the experimental evaluation, Eq. 3.5 is used, which defines the proposed method. The following settings have been used:

- **Values x and y** of the proposed method. x is a pre-defined value that is added to the similarity value returned from PCC and has been set 0.50 for the first(top) level, 0.375 for the second, 0.25 for the third and 0.125 for the last level. y is the similarity value returned from PCC and has been set to be greater than or equal to 0.33.
- **WPCC** recommendation method settings. For the MovieLens dataset this number has been set to 50. For the MovieTweetings and Epinions datasets has been set to 5.
- **MAE.** For the MAE evaluation, all the datasets have been split into two parts. An 80% randomly selected part that has been used for training and the remaining 20% used for testing. In the experimental results, *k* is the number of neighbors.
- **Precision and Recall.** For these metrics two experiments were conducted. The first one uses a five-user neighborhood and asks for the top 5 recommendations. The second one uses a ten-user neighborhood and asks for 10 recommendations. In the experimental results, *k* is the number of neighbors and *r* is the number of recommendations requested. It is noticed that, researchers state that measuring Recall is not practical in recommender systems and that Precision is more important in top N recommendations and the output values depend on the number of the recommendations requested and the size of the nearest neighborhood (Herlocker et al., 2004; Liu et al., 2014).
- **t1, t2, t3 and t4.** The values for these variables have been set to t1=50 t2=20 t3=10 t4=5. Based on function 5 the number of co-rated items for level 1 must be equal or greater than

50, level 2 is between 49 and 20, level 3 is between 19 and 10 and level 4 is between 9 and 5.

3.4.5 Results

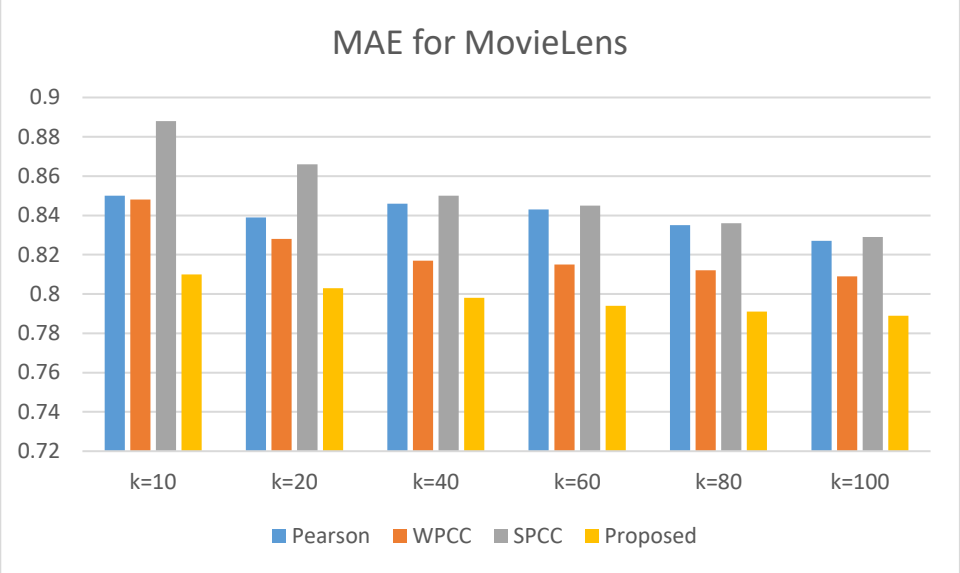


Figure 3-1 MAE results for the MovieLens dataset

The MAE results for the MovieLens 1m dataset are shown in Figure 3.1. It is shown that the proposed method ourperforms the other methods. Moreover, it is shown that as the number of neighbors is getting higher, the proposed method becomes more effective. It is also shown that the larger the neighbrhood grows the results become better for all methods.

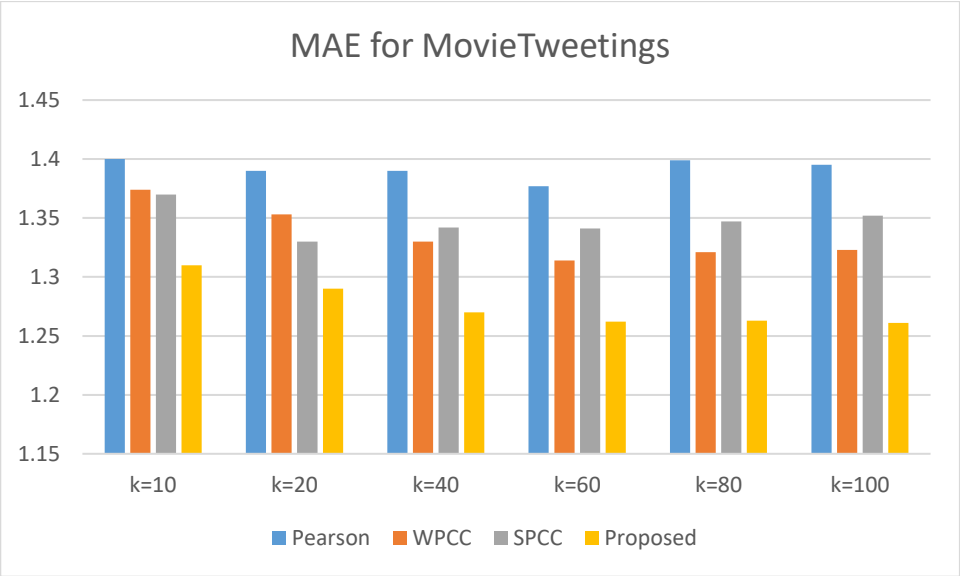


Figure 3-2 MAE results for the MovieTweetings dataset

The MAE results for the MovieTweetings dataset are shown in Figure 3.2. It is shown that the proposed method ourperforms the other methods. Furthermore, it is sown that as the number of neighbors is getting higher, the proposed method becomes more effective.

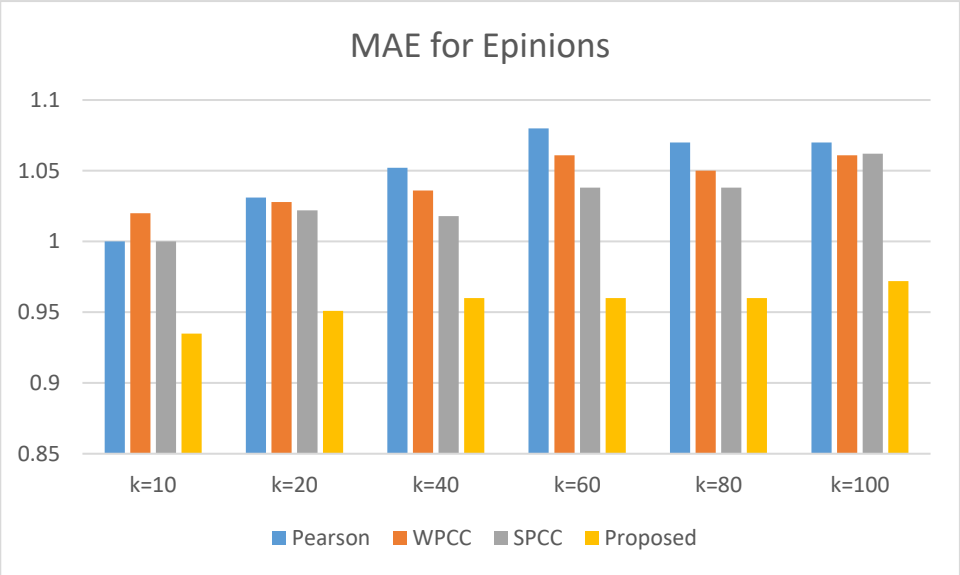
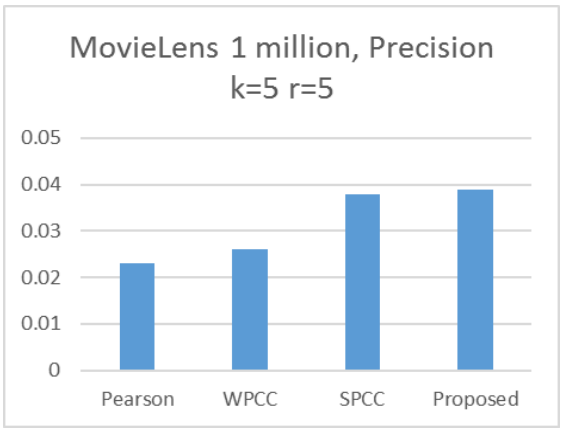
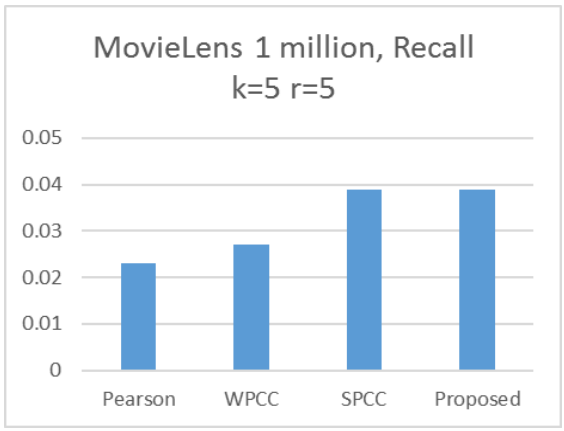


Figure 3-3 MAE results for the Epinions dataset

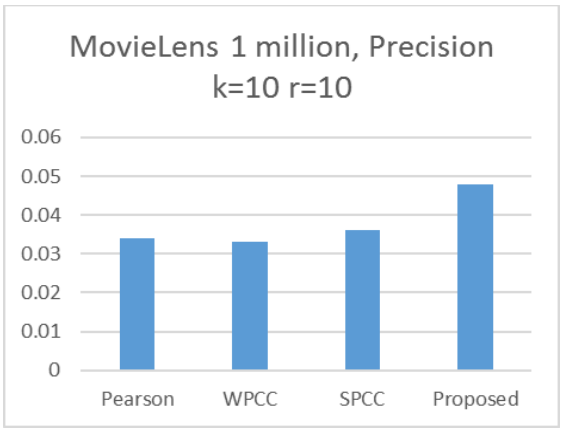
The MAE results for the Epinions dataset are shown in Figure 3.3. It is shown that the proposed method ourperforms the other methods. However in all cases there aren't any significant changes between the proposed method and the others when the size of the neighborhood grows.



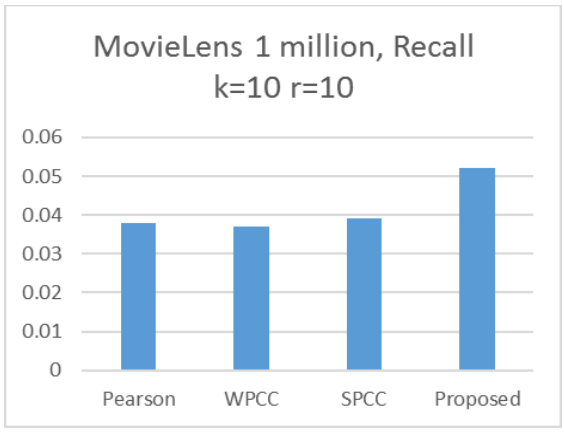
(a)



(b)



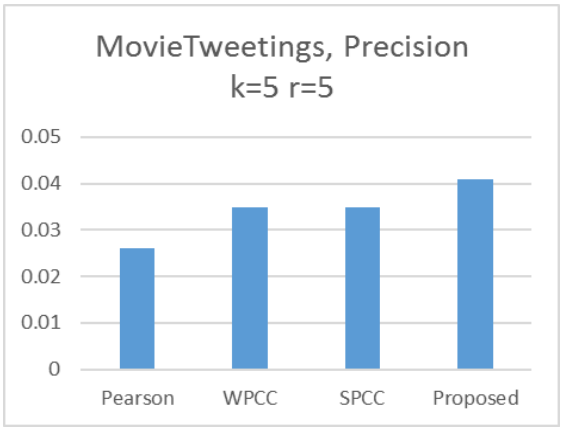
(c)



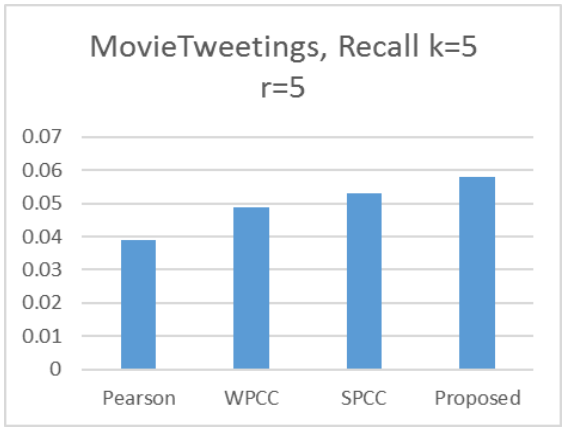
(d)

Figure 3-4 Precision and Recall results for the MovieLens 1 million dataset

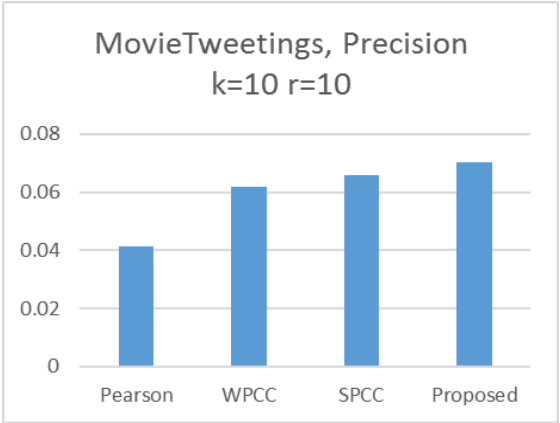
In Figure 3.4 the results obtained by using the Precision and Recall metrics using the MovieLens dataset are shown. It is shown that as the neighborhood grows larger, the quality of the top N recommendations becomes better.



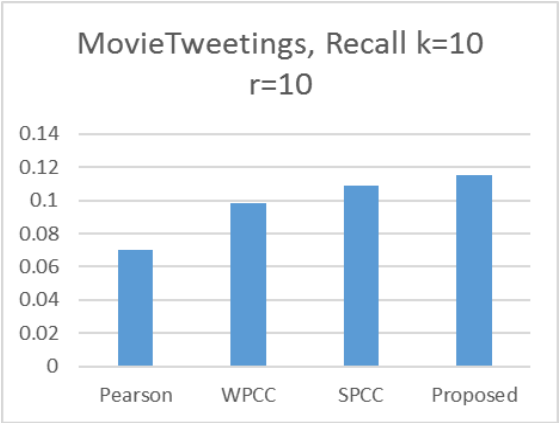
(a)



(b)



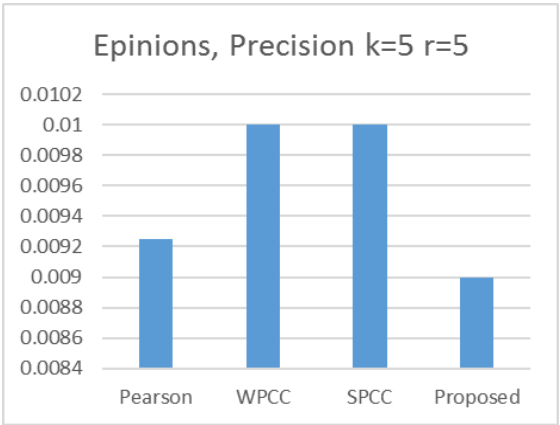
(c)



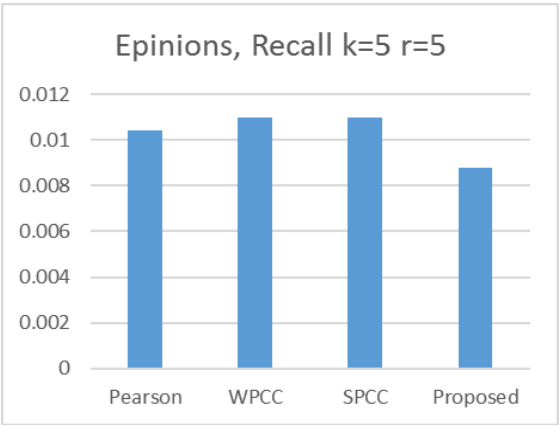
(d)

Figure 3-5 Precision and Recall results for the MovieTweetings dataset

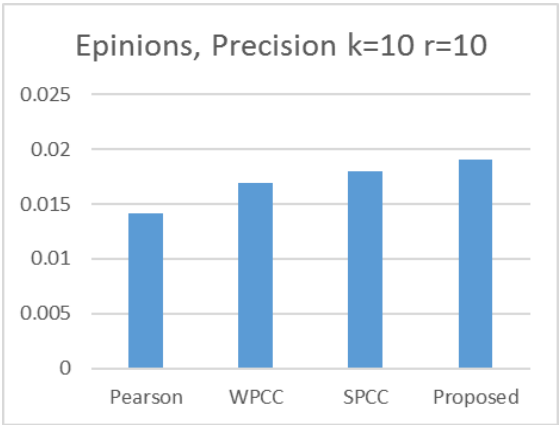
In Figure 3.5 the results obtained by using the Precision and Recall metrics using the MovieTweetings dataset are shown. It is shown that the proposed method produces results of better quality in all cases.



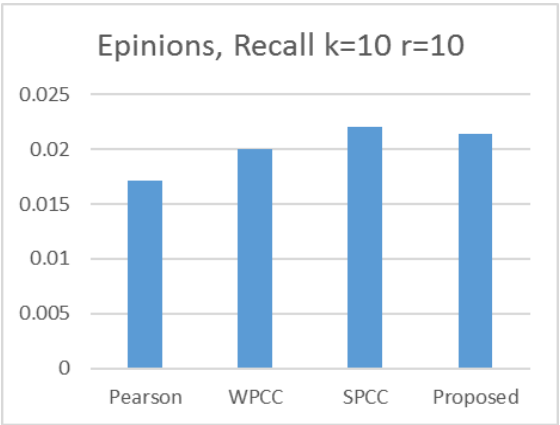
(a)



(b)



(c)



(d)

Figure 3-6 Precision and Recall results for the Epinions dataset

In Figure 3.6 the results obtained by using the Precision and Recall metrics using the Epinions dataset are shown. It is shown that as the neighborhood grows larger, the quality of the top N recommendations becomes better.

4 Dynamic multi-level collaborative filtering

One of the most used approaches for providing recommendations in various online environments such as e-commerce is collaborative filtering. Although, this is a simple method for recommending items or services, accuracy and quality problems still exist. Thus, the author proposes a dynamic multi-level collaborative filtering method that improves the quality of the recommendations. The proposed method is based on positive and negative adjustments and can be used in different domains that utilize collaborative filtering to increase the quality of the user experience. Furthermore, the effectiveness of the proposed method is shown by providing an extensive experimental evaluation based on three real datasets and by comparisons to alternative methods.

4.1 Introduction

Recommender systems are decision support systems found on the web in order to assist users about item or service selection, thus aiming to solve the information overload problem (Puglisi et al. 2015; Rodríguez-Hernández & Ilarri 2016). Moreover, collaborative filtering is the most widely used method for providing personalized recommendations in online environments such as e-commerce (Lu et al. 2015; Mishra et al. 2015; Jannach et al. 2010; Schafer et al. 2007; Liang 2008). In collaborative filtering systems, a database of user ratings is used and the generated recommendations are based on how much a user will like an unrated item based on his previous common rating history with other users. Thus, the recommendation process is based on an assumption about previous rating agreements, assuming that this agreement will be maintained in the future. Additionally, the ratings are used to create an n *x* m matrix with user ids, item ids and ratings, with an example of such a matrix shown in Table 4.1. This sample database has four users and four items with values from 1 to 5, while a ,-, denotes that a rating is not yet available for a particular item. The matrix is used as input when one of the users is requesting a recommendation and for a recommendation to be generated the degree of similarity between the user who makes the request and the other users' needs to be predicted using a similarity function such as the Pearson Correlation Similarity (PCC), which is defined in Eq. 4.1. At the next step a user neighborhood is created and it consists of users having the highest degree of similarity with the user who made the request. Finally, a prediction is generated after computing the average values of the nearest neighborhood ratings about an item, resulting in a recommendation list of items with the highest predicted rating values (Ekstrand et al. 2011; Jannach et al. 2010; Schafer et al. 2007).

	Item 1	Item 2	Item 3	Item 4
User 1	1	2	3	-
User 2	5	5	3	3
User 3	-	-	4	2
User 4	1	1	1	5

Table 4-1 A database of ratings

However, the accuracy of the predicted recommendations is still a major issue in collaborative filtering. Although, a simple similarity function such as PCC or Cosine can be used to provide recommendations there are still ways to improve the accuracy. A similarity function typically returns a value between -1 and 1, with -1 being the worst level of similarity between two users and 1 being the highest. In this method, it is shown that by modifying the similarity value between two users based on pre-specified constraints, the accuracy of the recommendations is improved. Moreover, the proposed method provides recommendations of better quality when compared to alternative methods. In addition, e-commerce vendors should be motivated to use the method because more accurate recommendations increase the chances of users buying products and the revenue of the business increases. Also, the search costs for the vendor are reduced and the user can find a relevant item easier, which results to less frustrated users

4.2 Related work

Collaborative filtering uses a database of ratings such as the one shown in Table 4.1. Furthermore it uses a similarity function such as PCC, defined in Eq. 4.1, where Sim (a,b) is the similarity value between two users a and b, also $r_{a,p}$ is the rating of user a for item p, $r_{b,p}$ is the rating of user b for item p and $\bar{r}a$ and $\bar{r}b$ represent the user's average ratings. P is the set of all items. Moreover, the similarity value ranges from -1 to 1 with higher values being better (Bobadilla et al. 2013; Ekstrand et al. 2011).

$$Sim_{a,b}^{PCC} = \frac{\sum p \in P(ra,p - \bar{r}a)(rb,p - \bar{r}b)}{\sqrt{\sum p \in P(ra,p - \bar{r}a)^2}\sqrt{\sum p \in P(rb,p - \bar{r}b)^2}} \quad (4.1)$$

A method that extends PCC by providing improved recommendations to users with a number of co-rated items that is above a predefined static threshold is the weighted PCC (WPCC) (Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl 1999) and is defined in Eq. 4.2. Furthermore, *Ia* and *Ib* represent the number of rated items from user a and b respectively.

$$Sim_{a,b}^{WPCC} = \begin{cases} \frac{|Ia \cap Ib|}{T} \cdot Sim_{a,b}^{PCC}, & if |Ia \cap Ib| < T \\ Sim_{a,b}^{PCC}, & otherwise \end{cases} \quad (4.2)$$

A similar approach to WPCC is SPCC (Jamali & Ester 2009) which is defined in Eq. 4.3. In this method users with smaller number of co-rated items have a weaker similarity. Furthermore, *Ia* and *Ib* represent the number of rated items from user a and b respectively.

$$Sim_{a,b}^{SPCC} = \frac{1}{1 + \exp(-|Ia \cap Ib|/2)} \cdot Sim_{a,b}^{PCC} \quad (4.3)$$

In (Koutrika et al. 2009), a similarity method called Jaccard is proposed that is based only on the number of co-rated items. Thus, the larger the number of the co-rated the similarity value is higher. In this method, the classical PCC method is not used. Other similarity approaches include the one proposed in (Lu et al. 2013), in which a fuzzy method is used to assign different weights to different sets of ratings. In (Wang et al. 2015), an entropy-based method is proposed, where the similarity values are calculated using a collaborative filtering function modified to use entropy. Furthermore, in (Liu et al. 2014), a similarity method is proposed that takes into consideration both the local and global user behavior. A different approach is offered by (Son 2014) that uses demographic data instead of user ratings in order to provide recommendations. In another similarity method found in the literature (Bobadilla, Ortega, Hernando & Bernal 2012) the authors aim to mitigate the new user cold start problem and to assist users with few ratings that face accuracy problems. In (Bobadilla, Ortega & Hernando 2012), a similarity method that is based on singularities is offered. This method derives contextual user information and uses it to calculate the singularity of each item. In (Bobadilla, Ortega, Hernando & Arroyo 2012), a balanced-based similarity measure is proposed that claims to provide high quality ratings with a low processing time. In (Ahn 2008), a similarity method is described that alleviates the cold start problem by proposing a heuristic method that improves the accuracy of collaborative filtering when few ratings are available. In (Anand & Bharadwaj 2011) a recommendation method that uses scarcity measures based on both local and global similarity values is proposed and claims to provide better recommendations when compared to traditional collaborative filtering. In (Polatidis & Georgiadis 2016), a recommendation method is proposed that is based on multiple-levels. In this method, the PCC similarity function is divided in different levels and if a PCC similarity belongs to a certain level is enhanced accordingly. In (Melville et al. 2002), a recommendation method is presented that uses content based information to enhance collaborative filtering recommendations. A method that is based on Jaccard and on

multidimensional vector spaces is proposed in (Sun, H. F., Chen, J. L., Yu, G., Liu, C. C., Peng, Y., Chen, G., & Cheng 2012) claiming to provide better recommendations when compared to the classical approaches. A different approach that uses a method to correct noisy ratings, thus providing better recommendations in overall (Toledo et al. 2015). In (Gan & Jiang 2013), the introduction of power law adjustments of user similarities is introduced, with the purpose of adjusting the similarity between users, thus having high accuracy and enhance the diversity of items. Moreover, collaborative filtering recommender systems can be based both on trust and distrust networks of users such as the one proposed in (Fang et al. 2015). These systems can either combine such information with the user rating network to provide more accurate recommendations to the users.

4.3 Proposed method

Different recommendation methods that cover a wide spectrum of approaches exist. the proposed method differs with the other methods since it is based on PCC and adds positive or negative adjustments based on information derived from PCC, such as the similarity value and the number of co-rated items. PCC and Cosine are the most widely used methods by collaborative filtering to provide recommendations (Konstan & Riedl 2012; Shi et al. 2014; Su & Khoshgoftaar 2009; Wang et al. 2015). The main objective is to adjust the similarity value provided by PCC, either in a positive or in a negative way. The proposed method is defined in Eq. 4.4 and its constraints are the number of co-rated items *T* and the similarity value provided by PCC. The author argues that if the number of co-rated items between two users is greater than or equal to a pre-specified threshold and the PCC value is greater than or equal to a pre-specified threshold then its similarity should be positively adjusted as shown in the first part of Eq. 4.4 and if one or both conditions do not hold then the second part of Eq. 4.4 is used to adjust the similarity value in a negative way.

$$Sim_{a,b}^{Proposed} = \begin{Bmatrix} Sim_{a,b}^{PCC} + Sim_{a,b}^{PCC}, if \frac{|Ia \cap Ib|}{T} \geq t \; and \; Sim_{a,b}^{PCC} \geq y \\ Sim_{a,b}^{PCC} * (\frac{1}{1 + (Sim_{a,b}^{PCC} * Sim_{a,b}^{PCC})}), \quad otherwise \end{Bmatrix} \quad (4.4)$$

The proposed method is based on the values provided by PCC and the co-rated items. In Eq. 4 *T* is the number of items that both user a and b have rated, *t* is the minimum number of co-rated items, y is a positive real number $y \in R$ and is the threshold PCC value. It is shown that with the use of such constraints, the accuracy of the provided recommendations is improved.

Although the proposed recommendation method improves the quality of the recommendations, a disadvantage of it is that the developer needs to alter the source code with static variable manipulation and execute several tests until the desired settings are derived. Thus, the author proposes the use of a dynamic multi-level method that is based on the proposed static method and in the method described in (Polatidis & Georgiadis 2017).

The author introduces the use of dynamic multiple-levels according to the characteristics available in the database. This method is simplified since it creates multiple levels according to the total number of users and items available in the database. Moreover, the use of the similarity value between users is not used in this part of the method. Subsequently, the similarity value between users is adjusted either positively or negatively using parts of Eq. 4.4.

Initially, Eq. 4.5 is used to derive the number of levels. The derived number is converted to the closest integer either by rounding up or down as necessary.

$$\mathrm{DvU} = \log_{10}(total\_no\_of\_users\_in\_dataset) \quad (4.5)$$

Then, Eq. 4.6 is used to come with the number of co-rated items that will be assigned to the first level. The first level is the higher as shown in Eq. 4.8 and will be adjusted positively with a higher value.

$$\mathrm{DvI} = \log_{2}(total\_no\_of\_items\_in\_dataset) \quad (4.6)$$

At the next step, the value from Eq. 4.6 is divided with the value from Eq. 4.5 as shown in Eq. 4.7.

$$\frac{DvI}{DvU} \quad (4.7)$$

This gives the number of co-rated items that users should have to be allocated to a certain level. Additionally, if the number is not an integer it is converted to its closest. However, a threshold of 5 is used as the minimum number of co-rated items for a similarity value to be adjusted positively and an extra level is dynamically created: a user who does not have the minimum number of co-rated items (as derived from Eq. 4.7) is allocated to this extra level. At this level, the similarity value is adjusted in a negative way. After the derivation of the number of levels and of the number of minimum and maximum co-rated items, Eq. 4.8 is used to adjust the similarity values between

users. Eq. 4.8 uses parts from Eq. 4.4 with modifications to adjust the similarities either in a positive or in a negative way.

$$Sim_{a,b}^{Proposed} = \begin{cases} Sim_{a,b}^{PCC} + Sim_{a,b}^{PCC}, & if \geq \frac{|Ia \cap Ib|}{M} \\ Sim_{a,b}^{PCC} + (\frac{Sim_{a,b}^{PCC}}{2}), & if \frac{|Ia \cap Ib|}{M} < m1 \; and \; \frac{|Ia \cap Ib|}{M} \geq m2 \\ Sim_{a,b}^{PCC} + (\frac{Sim_{a,b}^{PCC}}{3}), & if \frac{|Ia \cap Ib|}{M} < m2 \; and \; \frac{|Ia \cap Ib|}{M} \geq m3 \\ Sim_{a,b}^{PCC} + (\frac{Sim_{a,b}^{PCC}}{4}), & if \frac{|Ia \cap Ib|}{M} < m3 \; and \; \frac{|Ia \cap Ib|}{M} \geq m4 \\ \dots & \\ \dots & \\ \dots & \\ Sim_{a,b}^{PCC} + (\frac{Sim_{a,b}^{PCC}}{n}), & if \frac{|Ia \cap Ib|}{M} < mn1 \; and \; \frac{|Ia \cap Ib|}{M} \geq mn2 \\ Sim_{a,b}^{PCC} * (\frac{\frac{1}{1 + \left(Sim_{a,b}^{PCC} * Sim_{a,b}^{PCC}\right)}}{n} - 1), & otherwise \end{cases} \quad (4.8)$$

4.3.1 Application example

In this section, the author delivers an example with real numbers to make the application of the dynamic multi-level method easier to understand. For this example, data from the MovieTweetings dataset is used. This dataset has 39,363 users and 22,610 items.

The steps required to derive the levels are as follows:

1) Application of Eq. 4.5 to 39,363 gives DvU = 6.59
2) Rounding up of 4.59 gives 5
3) Application of Eq. 4.6 to 22,610 gives DvI = 14.46
4) Rounding down of 14.46 gives 14
5) Application of Eq. 4.7 gives $\frac{DvI}{DvU}$ = 14/5 = 2.8
6) Rounding up of 2.8 gives 3

Therefore, the number of levels that adjust the similarity in a positive way is 5 (although 4 levels are used for this dataset because the threshold of 5 co-rated items is used as a minimum requirement for the last level). In each level, the number of co-rated items is 3 and the threshold of the minimum co-rated items for the last level is 5. Moreover, an extra level is added that adjusts the similarity in a negative way. Algorithm 4.1 gives a detailed explanation of the levels created using the method.

Algorithm 4.1: Multiple levels derived from the MovieTweetings dataset

```
1)  If (n>=14) { /* Level 1, the Top/Higher level*/
2)  return (PCC+PCC);
3)  }
4)  else if (n>=11 && n<=13){ /* Level 2 */
5)  return (PCC)+(PCC/2);
6)  }
7)  else if (n>=8 && n<=10){ /* Level 3 */
8)  return (PCC)+(PCC/3);
9)  }
10) else if (n>=5 && n<=7){ /* Level 4 */
11) return (PCC)+(PCC/4);
12) }
13) else { /* Level 5 (Extra level) */
14) return (PCC*(1/1+(PCC)*(PCC)))/6;
15) }
```

Note that number 14 is assigned to the first level as shown in line 1 of the code: if two users have 14 or more co-rated items, the similarity gets the highest possible positive adjustment. Going down to level 2, it is shown that the number of co-rated items is either 11, 12 or 13 with that being 1-minus from the top level and the number of items in this level being 3. In level 3, again the range of co-rated items is 3 and in level 4 the same. Finally, level 5 is the negative adjustment.

4.4 Experimental evaluation

In this section, the experimental evaluation is explained, which was conducted on an Intel i3 2.13 GHz with 4GBs of RAM running windows 10 and the Java programming language. Furthermore, the experiments are based on three real datasets, MovieLens (Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl 1999), MovieTweetings (Dooms, S., De Pessemier, T., & Martens 2013) and Epinions (Massa & Avesani 2007).

The experimental evaluation contains accuracy comparisons based on the Mean Absolute Error (MAE), the Normalized Mean Absolute Error (NMAE), a 5-fold cross evaluation based on the Root Mean Square Error metric (RMSE), Precision, Recall, F1 and Hit rate measure comparisons (Herlocker et al. 2004; Schafer et al. 2007; Shani & Gunawardana 2011; Schröder et al. 2011; A. Bellogin et al. 2011).

4.4.1 Datasets

Three real datasets have been used to conduct the experiments. The datasets are MovieLens, MovieTweetings and Epinions.

- **MovieLens.** This is a real dataset obtained from the University of Minnesota which contains 4000 movies, 6000 users and 1,000,000 ratings. This is a publicly available dataset associated with the online movie recommendation of the university. The data are in the form of [userid] [itemid] [rating] and all the rating values are from 1 to 5.
- **MovieTweetings.** This is a real dataset crawled from twitter which contains 22610 movies, 39363 users and 431,780 ratings. The data are in the form of [userid] [itemid] [rating] and all the rating values are from 0 to 10. MovieTweetings is publicly available.
- **Epinions.** This is a real dataset crawled from Epinions.com website. It is a dataset for general, e-commerce, product recommendation and the data are in the form of [userid] [itemid] [rating] and all the rating values are from 1 to 5. The dataset has 664,824 ratings submitted from 49,290 users on 139,738 items. This dataset is publicly available.

These datasets have been used due the fact that are used in previous research, are from different domains and use different rating scales. The key statistics of the datasets are shown below in Table 4.2:

Dataset	Users	Items	Ratings	Sparsity	Rating range
MovieLens	6,000	4,000	1,000,000	0.958	1-5
MovieTweetings	39,363	22,610	431,780	0.999	0-10
Epinions	49,290	139,738	664,824	0.999	1-5

Table 4-2 Dataset statistics

4.4.2 Comparisons

The following recommendation methods have been used for the comparisons.

- **PCC.** This method is defined in Eq. 4.1 and is used to calculate the statistical correlation between two users based on common ratings to determine a similarity value from -1 to 1.
- **WPCC.** This method extends PCC by adding a constraint on the number of co-rated items. If that number is above the threshold, then the similarity is enhanced and otherwise PCC is used. Eq. 4.2 defines WPCC.
- **SPCC.** This method is also based on PCC. In this method, the similarity value returned for users with small co-rated items is weaker. SPCC is defined in Eq. 4.3.
- **Multilevel.** This is a method described in Polatidis & Georgiadis (2016) and it uses a static multilevel approach of 4 levels to provide recommendations based on collaborative filtering.
- **PLUS.** This is a method that is described in Gan & Jiang (2103). It uses a similarity method such as PCC and adjusts the similarity values between users using a power law function.

For this method, PCC is used and then the power law function is applied, as described in the first two steps of the research article.

In collaborative filtering although accuracy is an issue, however, more important is the quality of the Top-N (e.g. first 10 or first 20) recommendations which can be measured using Precision, Recall and F1. Thus, the first three and more established methods in measuring accuracy MAE, NMAE and 5-fold cross RMSE comparisons are used as benchmarks. Additionally, the quality comparisons are based on Precision, Recall and F1 measures and all the recommendation methods.

4.4.3 Measures

In order to measure the accuracy of the recommendations MAE has been used, which is a widely used metric for measuring accuracy (Herlocker et al. 2004; Schafer et al. 2007). MAE is defined in Eq. 4.9, where *pi* is the predicted rating and *ri* is the actual rating in the summation. MAE is used to measure the deviation between the predicted ratings and the actual ratings. Lower values are better. Additionally, normalized MAE (NMAE) has been used, which is a measure that is independent of the rating scale since its values are from 0 to 1, thus making the comparison between datasets easier. Eq. 4.10 defines NMAE (Shani & Gunawardana 2011; Ekstrand et al. 2011). In NMAE *rmax* is the maximum rating value available and *rmin* is the minimum rating value available in the rating scale (e.g. 1-5). Moreover, in NMAE lower values are better. Additionally, RMSE is defined in Eq. 4.11, where *pi* is the predicted rating and *ri* is the actual rating in the summation. In RMSE lower values are better.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|pi - ri| \quad (4.9)$$

$$NMAE = \frac{MAE}{rmax - rmin} \quad (4.10)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(pi - ri)} \quad (4.11)$$

However, in information retrieval systems use metrics such as Precision and Recall than can be used in recommender systems to measure the quality of the recommendations. In these metrics, a higher value is better. Eq. 4.12 defines Precision and Eq. 4.13 defines Recall. Moreover, Eq. 4.14

defines the F1 measure which combines Precision and Recall (Herlocker et al. 2004; Shani & Gunawardana 2011; Schröder et al. 2011; A. Bellogin et al. 2011). In Precision, Recall and F1 measures higher values are better. Eq. 4.15 defines hit rate, which is a classification measurement that reflects the proportion of users for which at least one recommendation can be made.

$$Precision = \frac{Correctly\ recommended\ items}{Total\ recommended\ items} \quad (4.12)$$

$$Recall = \frac{Correctly\ recommended\ items}{Relevant\ items} \quad (4.13)$$

$$F1 = 2 \cdot \frac{precision\ \cdot recall}{precision + recall} \quad (4.14)$$

$$hitrate_u = \begin{cases} 1 : if\ hits_u > 0 \\ 0 : else \end{cases} \quad (4.15)$$

4.4.4 Settings

For the experiments, the following settings have been used.

- **WPCC.** For the MovieLens dataset this number has been set to 50, for the MovieTweetings dataset this has been set to 10 and for the Epinions dataset this value has been set to 5. (Several experiments with the WPCC method have shown that the values used provide more accurate recommendations when compared to alternatives. Unfortunately, WPCC in its current form only takes static values).
- **MAE.** For the evaluation based on MAE as shown in Figure 4.1, Figure 4.2 and Figure 4.3 the datasets have been split into two parts. The first one is the training part consisting of 80% and the testing part consisting of the rest 20%. For the results in Figure 4.4, Figure 4.5 and Figure 4.6, 60% has been used for training and 40% for testing.
- **RMSE.** For the evaluation based on RMSE, the results of which are shown in table 4, the number of nearest neighbors has been set to 3 and a 5-fold-cross evaluation method has been used. In cross evaluations, the dataset is randomly sampled in *k* subsamples each of which has the same size. The evaluation of each dataset has been split into 5 subsamples. At the next step, 5 tests take place were each of the sample is used as a test model and the rest *K-1* subsamples are used as one training dataset. Thus, 5 different outputs and an average are provided as the results of this evaluation.

- The **t** and **y** values of the proposed (static) method have been set to 10 and 0.20 respectively for the MovieLens and MovieTweetings datasets. For the Epinions dataset the values have been set to 5 and 0.15 respectively. (Several experiments with the t and y values have shown that these values provide more accurate recommendations when compared to alternatives. Unfortunately, this is an issue when static methods are used).
- **PLUS.** This method is based on a function $f(x) = \alpha x^{\beta}$ where x is the similarity value between two users. Two tests took place with the value of α and β being 100 and 2 respectively in the first test, while in the second test the values 80 and 5 have been used, which provide lower quality results. (This is a static method and the set of numbers $\alpha = 100$ and $\beta = 2$ provided results of higher quality when after several tests).
- **hit rate.** To measure hit rate, Eq. 4.15 has been used to calculate the number of users that at least one recommendation can be provided to them. Then, the returned value is divided by the total number of users in the dataset and multiplied by 100 to deliver a percentage range.

4.4.5 Results

The MAE results obtained from the MovieLens dataset are shown in Figure 4.1. It is shown that the proposed method outperforms the alternatives and that as the user neighborhood grows then the method becomes more effective. In Figure 4.2 the results from the MovieTweetings dataset are shown. It is shown that the proposed method outperforms all the alternatives in every user neighborhood size. In Figure 4.3 the results from the Epinions dataset are shown. It is shown that the proposed method outperforms all the alternatives in every user neighborhood size, except the fisrt case. In all cases in the Figures and Tables k is the number of the nearest users forming the neighborhood.

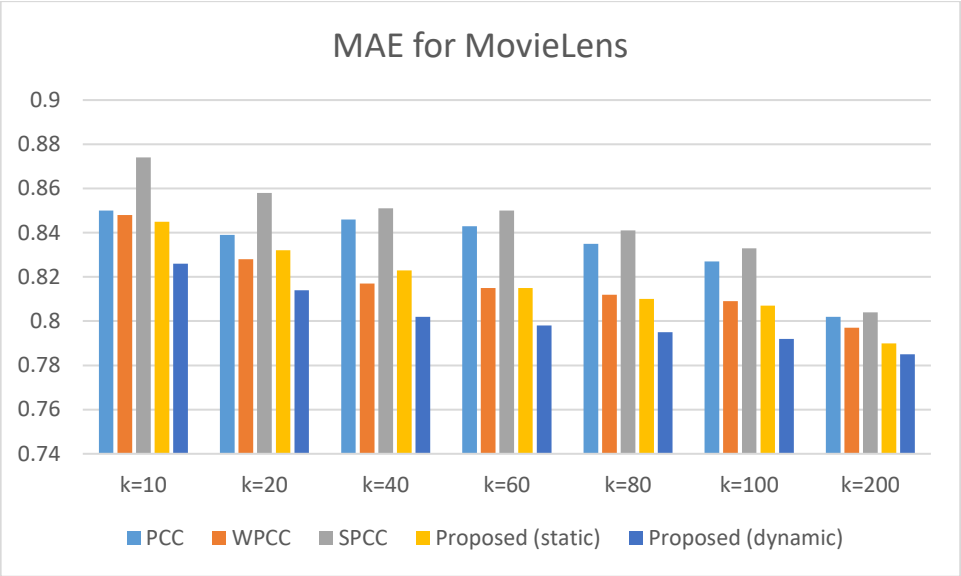


Figure 4-1 MAE results for MovieLens

Fig. 4.1 presents the results obtained from the MovieLens dataset. It is shown that the user neighborhood varies from 10 to 200 users. The proposed method, in its static and dynamic form, is compared against the traditional collaborative filtering recommendation methods in terms of MAE accuracy and clearly outperforms the alternatives. Moreover, good results are provided by WPCC, followed by PCC. SPCC does not work well with this dataset.

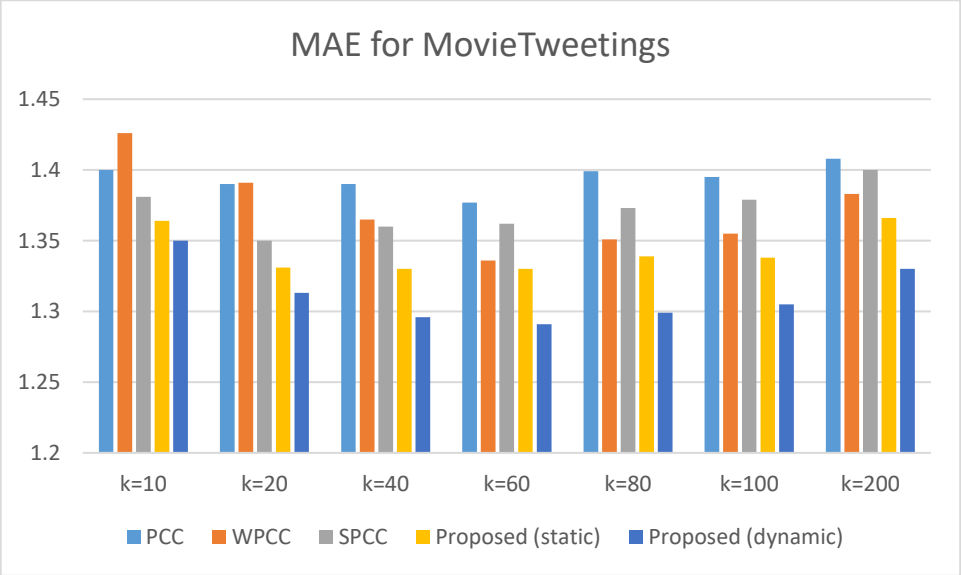


Figure 4-2 MAE results for MovieTweetings

Fig. 4.2 presents the results obtained from the MovieTweetings dataset. It is shown that the user neighborhood varies from 10 to 200 users. The proposed method, in its static and dynamic form, is compared against the traditional collaborative filtering recommendation methods in terms of

MAE accuracy and clearly outperforms the alternatives. Moreover, good results are provided by WPCC. Also, SPCC provides better results from the traditional PCC.

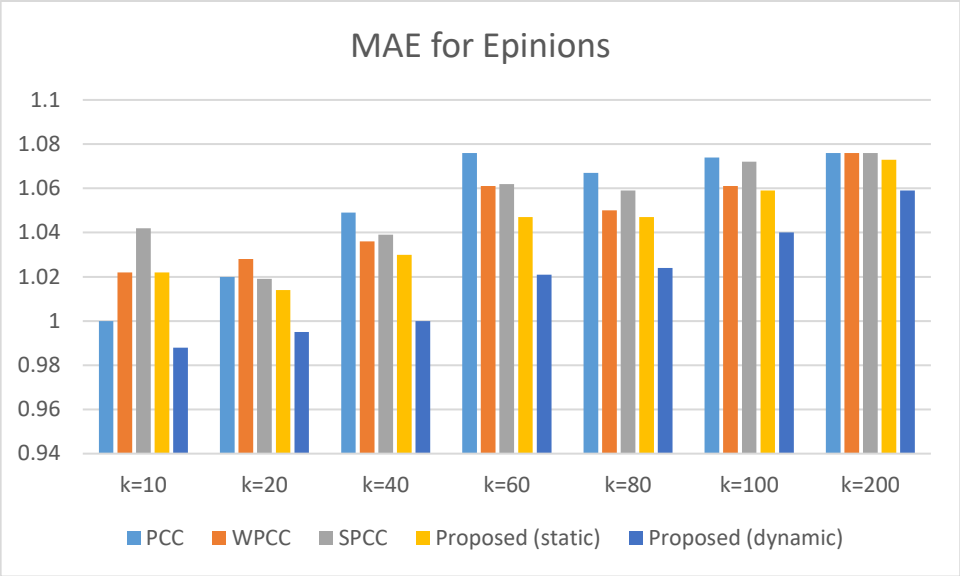


Figure 4-3 MAE results for Epinions

Fig. 4.3 presents the results obtained from the Epinions dataset. It is shown that the user neighborhood varies from 10 to 200 users. The proposed method, in its static and dynamic form, is compared against the traditional collaborative filtering recommendation methods in terms of MAE accuracy and clearly outperforms the alternatives. Moreover, WPCC and SPCC provide similar results as the neighborhood grows. Furthermore, in Figures 4.4, 4.5 and 4.6 show a comparison between the proposed method and PCC that is based on a 60% training set and a 40% test set, for MovieLens, MovieTweetings and Epinions. In these figures, the comparisons is about the traditional PCC and the proposed method in its static and dynamic form. The user neighborhood grows from 50 to 200 users and the proposed method outperforms PCC.

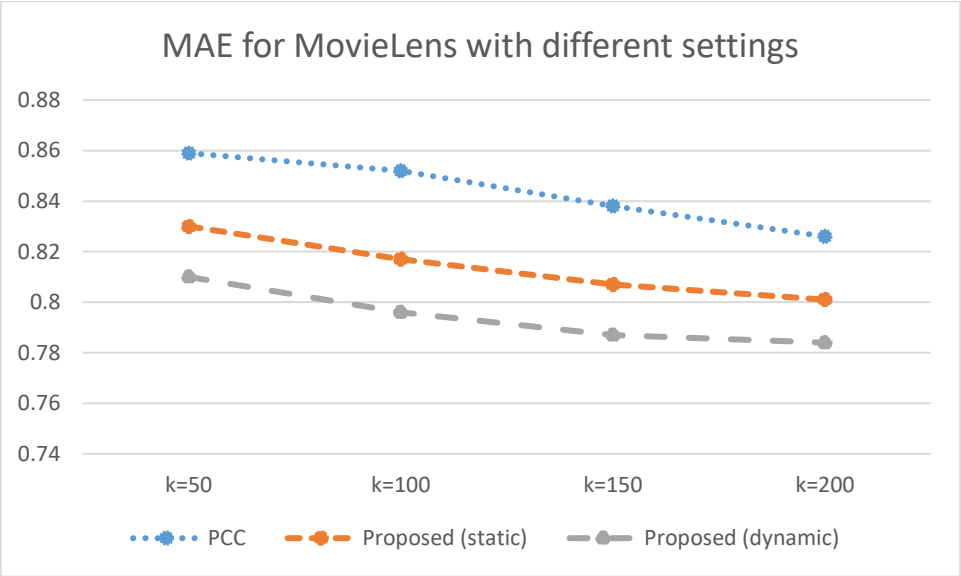


Figure 4-4 MAE results for MovieLens with different settings

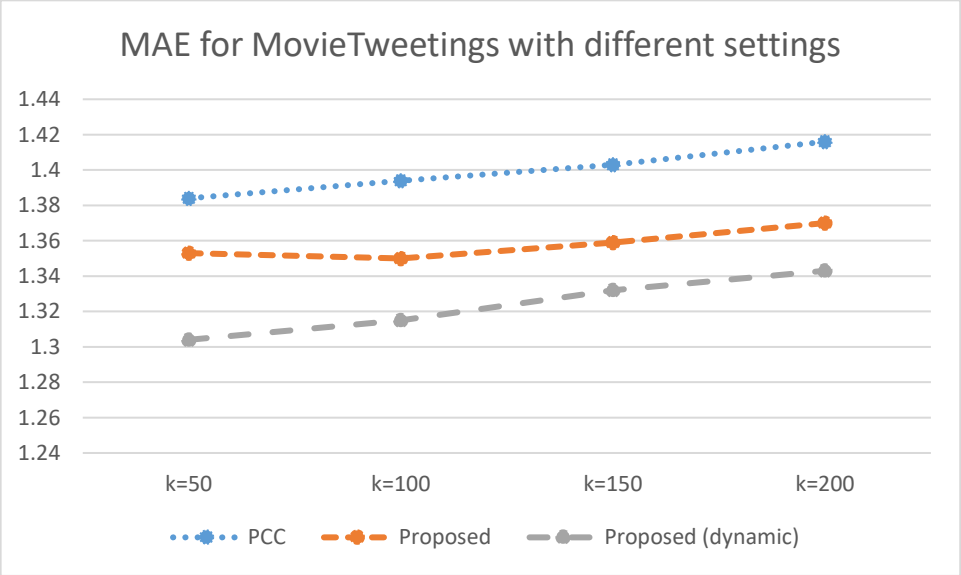


Figure 4-5 MAE results for MovieTweetings with different settings

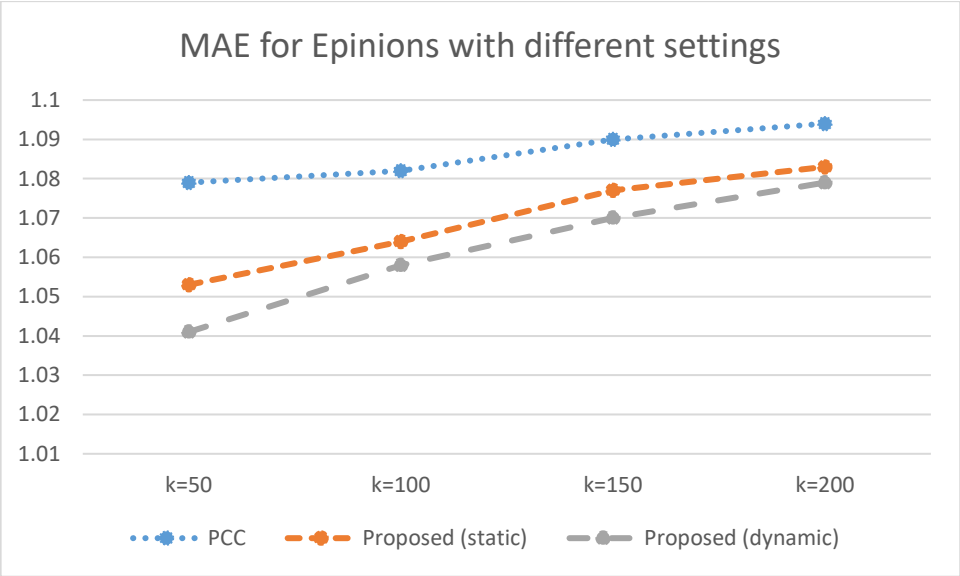


Figure 4-6 MAE results for Epinions with different settings

Additionnally in Tables 4.3 and 4.4 show a comparison using the NMAE measure, which makes comparison between different ratings scales, and therefore datasets, easier. In Table 4.3, k is set to 40 and in Table 4.4 to 80. The dataset is split to 80% for training and 20% for testing.

NMAE Comparisons

k=40

	MovieLens	MovieTweetings	Epinions
PCC	0.2215	0.139	0.2622
WPCC	0.2042	0.1365	0.2590
SPCC	0.2127	0.1360	0.2597
Proposed (static)	0.2057	0.1330	0.2575
Proposed (dynamic)	0.2005	0.1296	0.2500

Table 4-3 NMAE comparison with k=40

NMAE Comparisons

k=80

	MovieLens	MovieTweetings	Epinions
PCC	0.2087	0.1399	0.2667
WPCC	0.2030	0.1351	0.2625
SPCC	0.2102	0.1373	0.2647
Proposed (static)	0.2025	0.1339	0.2617
Proposed (dynamic)	0.1987	0.1299	0.2560

Table 4-4 NMAE comparison with k=80

Table 4.5. lists the results obtained from the 5-fold cross validation tests based on the RMSE metric and with comparisons to the traditional methods. In this experiment the datasets have been split to 5 nearly equal sized parts where each part is used as a traning for the next part which is the testing part. This takes places until 5 tests are reached. The number of neighbors has been set to 3. Moreover, five tests have taken place and an average has been calculated. It is shown that the proposed method can outpeform the traditional collaborative filtering methods in terms of RMSE, cross-fold validation and with a small user neighborhood.

	5-fold cross validation based on RMSE					
Dataset	5 fold cross-validation	PCC	WPCC	SPCC	Proposed (static)	Proposed (dynamic)
MovieLens	1	1.146	1.102	1.139	1.102	1.092
	2	1.110	1.095	1.134	1.077	1.075
	3	1.130	1.095	1.128	1.071	1.093
	4	1.144	1.099	1.147	1.078	1.083
	5	1.105	1.084	1.144	1.076	1.084
	Average	1.127	1.095	1.138	1.081	1.085
MovieTweetings	1	1.907	1.898	1.849	1.844	1.800
	2	1.908	1.897	1.850	1.818	1.773
	3	1.858	1.855	1.825	1.777	1.777
	4	1.869	1.858	1.850	1.846	1.778
	5	1.874	1.867	1.848	1.827	1.752
	Average	1.883	1.875	1.844	1.822	1.776
Epinions	1	1.373	1.316	1.370	1.337	1.261
	2	1.384	1.306	1.316	1.279	1.273
	3	1.362	1.303	1.397	1.364	1.299
	4	1.380	1.312	1.382	1.357	1.270
	5	1.358	1.292	1.352	1.328	1.273
	Average	1.372	1.306	1.363	1.333	1.275

Table 4-5 RMSE 5-fold cross validation results

Table 4.6 presents a comparison of the proposed method and alternatives. The comparisons are based on Precision, Recall and F1. The number of nearest neighbors, *k*, is set to 150 and the number fraof the requested recommendations *r* is set to 20. It is shown that the proposed method in its static form produces results marginally better than PCC, WPCC and SPCC. Furthermore, the proposed method in its dynamic form produces results that are better than all the method in most cases. However, note that the proposed (dynamic) method perform slightly lower in the MovieLens dataset when compared to the state-of-the-art but better in MovieTweetings and Epinions datasets

Precision, Recall and F1 comparisons of the proposed static and dynamic methods with alternatives
k=150, r=20

	MovieLens			MovieTweetings			Epinions		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
PCC	0.106	0.124	0.114	0.101	0.190	0.130	0.032	0.041	0.036
WPCC	0.142	0.160	0.145	0.114	0.209	0.142	0.038	0.048	0.042
SPCC	0.102	0.120	0.108	0.103	0.195	0.134	0.031	0.041	0.035
Multilevel	0.180	0.209	0.193	0.126	0.235	0.160	0.039	0.049	0.043
PLUS α=100 β=2	0.174	0.202	0.187	0.120	0.225	0.157	0.038	0.050	0.043
PLUS α=80 β=5	0.153	0.179	0.165	0.117	0.220	0.153	0.037	0.048	0.041
Proposed (static)	0.143	0.167	0.148	0.116	0.217	0.150	0.037	0.048	0.041
Proposed (dynamic)	0.173	0.202	0.181	0.127	0.237	0.164	0.043	0.056	0.048

Table 4-6 Precision, Recall and F1 comparisons

Table 4.7 presents the hit rate comparison between the proposed method and alternatives. The number of nearest neighbors, *k,* is set to 150 and the number of the requested recommendations *r* is set to 20. Furthermore, at least one recommendation needs to be provided to a user to have a hit rate value greater than zero for this user. Then, the output represents the overall value between all users. It is shown that the proposed dynamic method and the multilevel method can produce the same number of recommendations for the MovieLens dataset. For the MovieTweetings dataset the results are similar with the plus method producing the same results and in the Epinions dataset the proposed (dynamic) method can produce more recommendations

Hit rate comparisons of the proposed static and dynamic methods with alternatives
k=150, r=20

	PCC	WPCC	SPCC	Multilevel	PLUS α=100 β=2	PLUS α=80 β=5	Proposed (static)	Proposed (dynamic)
MovieLens	50%	52%	50%	57%	56%	55%	55%	57%
MovieTweetings	6.6%	6.7%	6.7%	6.9%	6.9%	6.8%	6.8%	6.9%
Epinions	1.2%	1.3%	1.2%	1.2%	1.3%	1.3%	1.3%	1.4%

Table 4-7 Hit rate comparisons

5 Random perturbation rating privacy

Collaborative recommender systems offer a solution to the information overload problem found in online environments such as e-commerce. The use of collaborative filtering, the most widely used recommendation method, gives rise to potential privacy issues. In addition, the user ratings utilized in collaborative filtering systems to recommend products or services must be protected. The purpose of the author is to provide a solution to the privacy concerns of collaborative filtering users, while maintaining high accuracy of recommendations. The author proposes a multi-level privacy-preserving method for collaborative filtering systems by perturbing each rating before it is submitted to the server. The perturbation method is based on multiple levels and different range of random values for each level. Before the submission of each rating, the privacy level and the perturbation range is selected randomly from a fixed range of privacy levels. Furthermore, the proposed privacy has been experimentally evaluated method using five real datasets and show that with a small decrease of utility, user privacy can be protected, while the proposed approach offers practical and effective results.

5.1 Introduction

Recommender systems aim to solve the information overload problem found in various online environments such as e-commerce and social networks (Polatidis & Georgiadis 2016; Shi et al. 2014; Moradi & Ahmadian 2015). Such systems can be used to recommend products, services, or users to users both implicitly or explicitly. The most used recommendation method is collaborative filtering (Shi et al. 2014; Ekstrand et al. 2011).

Collaborative filtering is a recommendation method where a database of ratings is utilized and a similarity measure is used to make predictions based on ratings provided by other registered users (Shi et al. 2014; Konstan & Riedl 2012). A database of ratings is essential and an example of such is shown in Table 5.1.

	Product 1	Product 2	Product 3
User 1	3	2	-
User 2	-	5	4
User 3	3	4	4
User 4	4	3	3

Table 5-1 A ratings database

Considering that a user wants recommendations, a similarity function such as the Pearson Correlation Similarity, is used to create a neighborhood of the most similar users for the user who is requesting the recommendations. Pearson calculates the statistical correlation between two users and returns a value from -1 to 1. Pearson correlation is shown in Eq. 5.1 and is the most frequently method used in collaborative filtering (Ekstrand et al. 2011). P is the set of all products, Sim (a, b) is the similarity between two users a and b, $r_{a,p}$ is the rating of user a for product p, $r_{b,p}$ is the rating of user b for product p, and $\bar{r}a, \bar{r}b$ represent the users' average ratings.

$$Sim\ (a,b) = \frac{\sum p \in P(r\ (a,p) - (\bar{r}a))(r\ (b,p) - (\bar{r}b))}{\sqrt{\sum p \in P(\ r\ (a,p) - (\bar{r}a))^2}\sqrt{\sum p \in P(\ r\ (b,p) - \ (\bar{r}b))^2}} \quad (5.1)$$

The recommendations are generated based on rating predictions on how likely a user is to prefer items that they have not looked at yet. These are based on historical ratings common with those of other users. The algorithm will compute the degree of similarity using a function such as that of Pearson between the user who is requesting the recommendations and those of other users. A neighborhood of the most similar users is created consisting of the ones with the higher degree of similarity. As a last step, rating predictions for unobserved items are generated between previous preferences of the user and those of the neighbors and the items with the highest rating predictions are recommended.

Privacy is an important issue for users of such systems (Bilge & Polat 2013; Ozturk & Polat 2015). In the context of ratings, privacy concerns make users unwilling to submit ratings, thus leading to sparsely populated relevant datasets which in turn can lead to lower degrees of similarity and eventually to poor recommendations. In typical scenarios, a recommendation system which is based on the client-server model, accepts requests from users and responds with recommendations. The ratings are submitted directly from the client to the server. Users need to be registered in order to receive personalized recommendations. As submitted ratings are directly linked to individual users, the privacy of the ratings is considered an important aspect (Berkovsky et al. 2012; Jeckmans et al. 2013; Toch et al. 2012; Kobsa 2007). In the context of the work, privacy is particularly related to the protection of the ratings. To avoid data leakage, and given the fact that the ratings are one of the most important information found in collaborative filtering these should be protected by using an appropriate privacy-preserving system (Kobsa 2007). The perturbation of ratings is often utilized to achieve the desired level of privacy. The two most common cases where perturbation of ratings is essential in ensuring privacy are: a) data release, where a subset of stored ratings is transferred to a user's personal computer / device for local processing and b) where employees could be in a position of exploiting their access rights to registered users' private information.

5.2 Related work

In collaborative filtering, there are privacy concerns about user ratings, which can be collected by the service provider or untrusted third-parties. Thus, users may not be willing to submit ratings or might submit fake ratings, thus, resulting in recommendations with poor relevance. Thus, generating as accurate recommendations as possible, while preserving user privacy is a serious challenge. According to (Lam et al. 2006), there are three main threats that may cause a collaborative filtering method not to work as expected and are the following:

1. The undesired access to private data by untrusted parties.
2. The manipulation of private user profiles to recommend certain products or services.
3. The service denial or malfunctioning of the system.

The two main approaches that can be utilized for privacy-preservation of personal user data such as the ratings are:

Centralized: Where all data are stored in a single server.
Decentralized: Where the data are distributed in more than one location and/or server.

In the case of centralized approach, there are several different methods for privacy-preserving recommendations:

A classical approach for privacy-preserving collaborative filtering is that of rating modification. A randomized perturbation technique, which perturbs every rating before it is submitted to the server has been developed by (Polat & Du 2005). In their method, the perturbation value is derived from a distribution. Another privacy-preserving collaborative filtering approach that is based on a bisecting k-means algorithm is proposed by (Bilge & Polat 2013). In this approach the authors propose a preprocessing scheme that is based on two stages. Initially the algorithm uses a binary decision tree while in the second step it creates clones of users by injecting pseudo predictions in the original user data. Kikuchi and Mochizuki, proposed a method for privacy-preserving collaborative filtering that adds random noise to the original rating data and then uses a posterior probability distribution method based on Bayes for reconstructing the original distribution of ratings (Kikuchi & Mochizuki 2012). An interesting approach that uses data obfuscation to provide privacy-preserving collaborative filtering is found in (Parameswaran & Blough 2007). In this method, recommendations are generated by combining data from multiple sources and obfuscating them before sending them to a centralized database. Additional works based on data modification include the one offered by (Zhang et al. 2006). In this work, an agreement is established between

the server and the users regarding the disclosure measure and the server sends guidelines to the user for modifying the data before submission. A different approach offering to protect ratings in a form of k-anonymity can be found in (Casino et al. 2015). A number of *k* clusters of users is created, with each cluster having the same ratings and value for each of the clusters. This method is used when all the ratings are available in a centralized database and need to be released. In (T. Zhu et al. 2014) differential noise is used to protect user privacy by providing nearest neighborhood attack resistance. The noise is added at the produced similarity value to avoid attacks from people who observe the generated recommendations and thus, can guess what ratings to submit to affect the generated output. An interesting approach has been proposed by (Zhang et al., 2014). In this method, the authors use a combination of a uniform and Gaussian distribution to perturb a rating before it is submitted to the server. After the submission of the perturbed rating an intensity weight is also submitted to the server. This intensity weight assists the server to produce results of higher accuracy when compared to simple perturbation methods.

An alternative direction in providing privacy-preserving recommendations is through the means of distributed storage, where attackers need to gain access to multiple databases instead of centralized one and the guidelines for generating recommendations using forms of distributed computing are outlined in (Tveit 2001). In "PocketLens", (Miller et al. 2004) show that the performance of distributed collaborative filtering is close to that of centralized systems. In (Canny 2002) a method it proposed where the users have control of their data and are grouped into communities. In this method, when recommendations were requested, all the data of people who form the community was combined into one output and individuals are more protected. In (Aimeur et al. 2008) a cryptographic approach is used when the client is communicating with the server and the use of a semi-trusted third party is proposed. In (Shokri et al. 2009) a method is proposed that is used to enlarge a user profile with other similar profiles, using a distributed mechanism, before sending any data to the server. In this approach, every user stores her profile offline and merges it partly with profiles of similar users after direct contact with them. An alternative decentralized approach can be found in (Kaleli & Polat 2010), where a community of people is used to create a peer to peer network.

In addition to these two categories of approaches, other relevant privacy-preserving methods exist and include techniques that have been listed in (Kobsa 2007) and(Toch et al. 2012). These make use of pseudonymous and anonymous user modelling, client-side personalization and encrypted aggregation. A rather interesting approach is that of personalization of privacy, where each user has a personalized plan regarding his privacy level, which results to different perturbation levels (Kobsa 2007).

Although, the related works are interesting and add their unique values to the literature, the proposed method described in section 3 adds its own characteristics by introducing randomization levels. Furthermore, multiple levels in privacy-preserving collaborative filtering can be used by other methods as well. Multi-level privacy brings an extra level of confusion to potential attackers since it makes it harder to guess the real rating value. Furthermore, attackers or people who can gain access to the database of ratings in order to affect the output by executing a KNN attack is more difficult, since (a) the rating values are perturbed and (b) future submitted ratings will be perturbed after a two-step randomization algorithm takes place at the client side.

5.3 Proposed method

The author proposes a centralized approach, focused on rating privacy. The method is based on the personalization of privacy, and from that point of view the approach is influenced by the work of (Kobsa 2007), that also offers a solution that uses different perturbation levels. In the method the privacy level and, thus, the perturbation range is created randomly for each submitted rating. Therefore, each perturbed rating discloses no information about the actual perturbation range that has been used, thus making it harder to guess the real value. Thus, the author proposes the use of a multi-level method for privacy-preserving collaborative filtering (Polatidis, Georgiadis, Pimenidis & Mouratidis 2017). In the privacy-preserving recommendation system the insertion of a random value at the actual rating before it is submitted to the server is utilized. Randomization is a widely-used privacy preservation method in privacy-preserving data mining (Aggarwal & Yu 2008). Furthermore, the initiative of using a simple logic algorithm for perturbation is that it can be easily installed in the client side and that is less complex to use and understand. A high-level overview of the approach is shown in Figure 5.1.

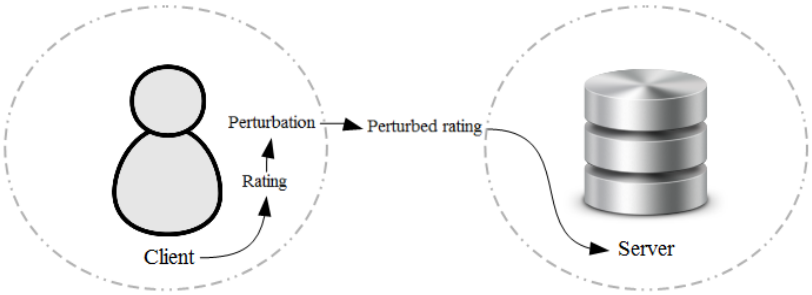


Figure 5-1 Privacy-preserving rating submission

In the proposed method, the insertion of multiple privacy levels (with each level perturbing the rating with a different range of values), sufficiently protects user privacy while maintaining a

decent level of accuracy. Algorithm 5.1 describes the proposed perturbation method. The values of MINRATING and MAXRATING refer to the rating scale used by the recommender system. For example, in the case that a scale in the range 1 to 5 is used then MINRATING refers to 1 and MAXRATING refers to 5. If a value, due to the perturbation method, drops below MINRATING then the perturbed rating takes the value of MINRATING and in the case, that it exceeds MAXRATING then the perturbed rating takes the value of MAXRATING.

Algorithm 5.1: Privacy (Runs on the client)

Input: Rating
Output: PerturbedRating

```
Integer Level = Generate Random [1…n] // Privacy levels from 1=low, to 2=medium to 3=high, to n=higher
Integer Rand // Random integer [-t…t]
If (Level=1)
Generate random value Rand from [-1…1] // -1, 0 or 1
Perturbed Rating = Rating + Rand
        If (PerturbedRating < MINRATING)
                PerturbedRating = MINRATING
        Else If (PerturbedRating >MAXRATING)
                PerturbedRating = MAXRATING
        End If
End If
Else If (Level=2)
Generate random value Rand from [-2…2] // -2, -1, 0, 1 or 2
Perturbed Rating = Rating + Rand
        If (PerturbedRating < MINRATING)
                PerturbedRating = MINRATING
        Else If (PerturbedRating >MAXRATING)
                PerturbedRating = MAXRATING
        End If
End Else If
Else If (Level=3)
Generate random value Rand from [-3…3] // -3, -2, -1, 0, 1, 2 or 3
Perturbed Rating = Rating + Rand
        If (PerturbedRating < MINRATING)
                PerturbedRating = MINRATING
        Else If (PerturbedRating >MAXRATING)
                PerturbedRating = MAXRATING
        End If
End Else If
Else If (Level=n)
Generate random value Rand from [-n…n] // -n, -3, -2, -1, 0, 1, 2, 3n
Perturbed Rating = Rating + Rand
        If (PerturbedRating < MINRATING)
                PerturbedRating = MINRATING
        Else If (PerturbedRating >MAXRATING)
                PerturbedRating = MAXRATING
        End If
End Else If
Return PerturbedRating
```

5.4 Experimental evaluation

In this section, the method is experimentally evaluated using five real datasets and widely used metrics with different parameters. The experiments were conducted on an Intel i3 2.13 GHz with 4GBs of RAM, running Linux. All the algorithms have been implemented using the Java programming language and the Recommender101 library (Jannach et al. 2013).

5.4.1 Real datasets

For the evaluation of the psoposed privacy-preserving method, different experimentation settings have been used along with five real datasets which are publicly available and widely used in evaluating recommenders. The datasets are MovieLens (Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl 1999), MovieTweetings (Dooms, S., De Pessemier, T., & Martens 2013), YahooMovies, YahooAudio and FilmTrust (Guo, G., Zhang, J., & Yorke-Smith 2013). Moreover, Table 5.2 presents the basic statistics of the datasets.

- **MovieLens**

MovieLens is an online movie recommender system. The dataset contains 100 thousand ratings, with values from 1 to 5, of 1,682 movies from 943 users. The data have been collected from the University of Minnesota from their online movie recommender system.

- **MovieTweetings**

MovieTweetings is a dataset that has been crawled from Twitter users and is publicly available. The rating scale of the dataset is in the range 0 to 10. The dataset contains 431,780 ratings from 39,363 users for 22,610 items.

- **YahooMovies**

YahooMovies is a dataset obtained from Yahoo!, which is a part of the Yahoo! Webscope program. The rating scale of the dataset is in the range 1 to 13. The dataset contains 211,231 ratings, 11,915 movies and 7,642 users.

- **YahooAudio**

YahooAudio is a dataset obtained from Yahoo!, which is a part of the Yahoo! Webscope program. The rating scale of the dataset is in the range 1 to 5. The dataset contains 311,704 ratings, 1,000 songs and 15,400 users.

- **FilmTrust**

FilmTrust is a dataset crawled from the FilmTrust website. The rating scale of the dataset is in the range 0,5 to 4. The dataset contains 35,497 ratings, 2,071 movies and 1,508 users.

Dataset	Users	Items	Ratings	Rating scale
MovieLens	943	1,682	100,000	1 - 5
MovieTweetings	39,363	22,610	431,780	0 - 10
YahooMovies	7,642	11,915	211,231	1 - 13
YahooAudio	15,400	1,000	311,704	1 - 5
FilmTrust	1,508	2,071	35,497	0,5 - 4

Table 5-2 Dataset statistics

5.4.2 Accuracy measures

For measuring the accuracy of the generated recommendations of the proposed method the author has used the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). These metrics are widely accepted for evaluating recommender systems (Herlocker et al. 2004; Shani & Gunawardana 2011; Jannach et al. 2010). MAE is defined in Eq. 5.2 with *pi* being the predicted rating and *ri* being the actual rating in the summation. MAE is used for computing the deviation between the predicted and the real ratings. Note that lower values mean better recommendation predictions. RMSE is shown in Eq. 5.3. RMSE is an equation that is like MAE but with squared values. In RMSE, lower values are better.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|pi - ri| \quad (5.2)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(pi - ri)} \quad (5.3)$$

5.4.3 Settings

For the experiments, the following settings have been used:

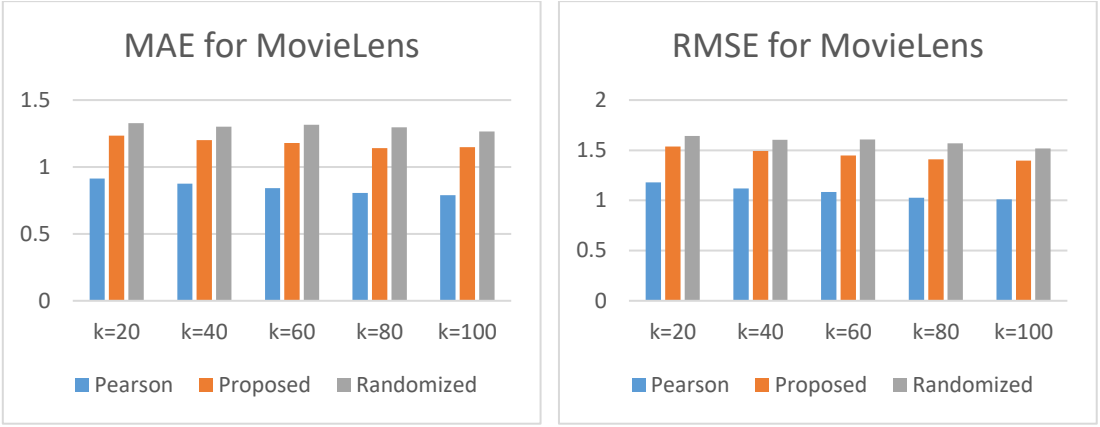
- **MovieLens dataset.** For this dataset, due to the rating scale being from 1 to 5 only the first two levels of the proposed method have been used.
- **MovieTweetings dataset.** For this dataset, the proposed method has used three levels.

- **YahooMovies.** For this dataset, four levels of the proposed method have been used.
- **YahooAudio.** For this dataset, the proposed method has used two levels.
- **FilmTrust.** For this dataset, the first two levels have been used.
- **MAE and RMSE.** For MAE and RMSE a 10-fold cross validation method has been used across all tests.
- **Pearson.** This is produced using collaborative filtering, Eq. 5.1 and the unaltered datasets.
- **Proposed method.** This is produced using collaborative filtering, Eq. 5.1 and the modified datasets having used the proposed method.
- **Randomized perturbations:** In this method, every rating is perturbed from a fixed range of values. For the MovieLens dataset this is from -2 to 2, for the MovieTweetings dataset this has been set from -3 to 3, for the YahooMovies dataset this is from -4 to 4, for the YahooAudio this is from -2 to 2 and for the FilmTrust dataset from -2 to 2. The fixed range of values for this algorithm have been selected because they are the same values used in the levels of the proposed method. This method is discussed in (Berkovsky et al. 2012).

However, it should be noted that the range of random value insertion has been chosen after experimentation with different values. The selected fixed values provided the best results between accuracy and privacy protection for each of the datasets. Although, different fixed ranges or random ranges of numbers can be used in all methods if the perturbed rating is within the scale used by the recommender system. In any case experiments need to take place in order to verify that accurate recommendations can still be provided.

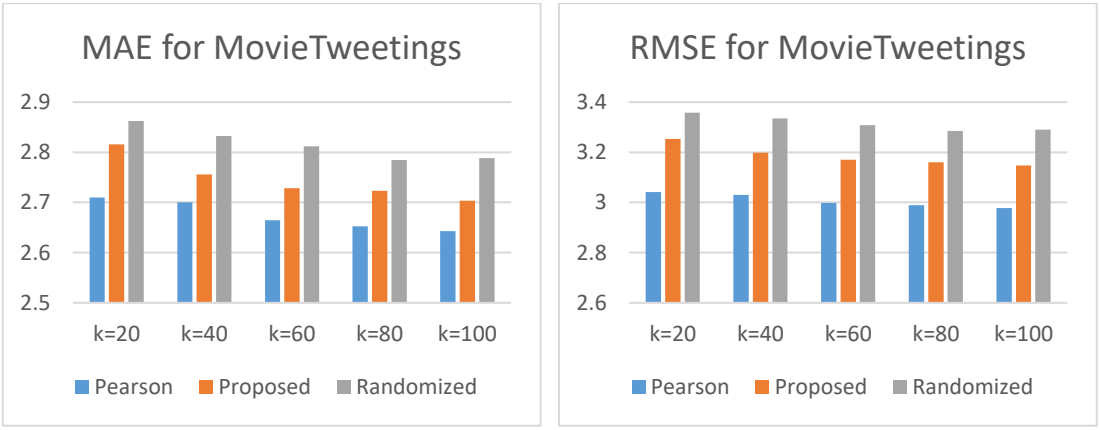
5.4.4 Results

The MAE and RMSE results obtained from the MovieLens dataset are shown in Figure 5.2. In Figure 5.3 the results from the MovieTweetings dataset are shown. In Figures 5.4 and 5.5 the results from the YahooMovies and YahooAudio results are shown and in Figure 5.6 the results from the FilmTrust dataset are shown. In all Figures, sub-Figure (a) represents the MAE results and sub-Figure (b) represents the RMSE results.



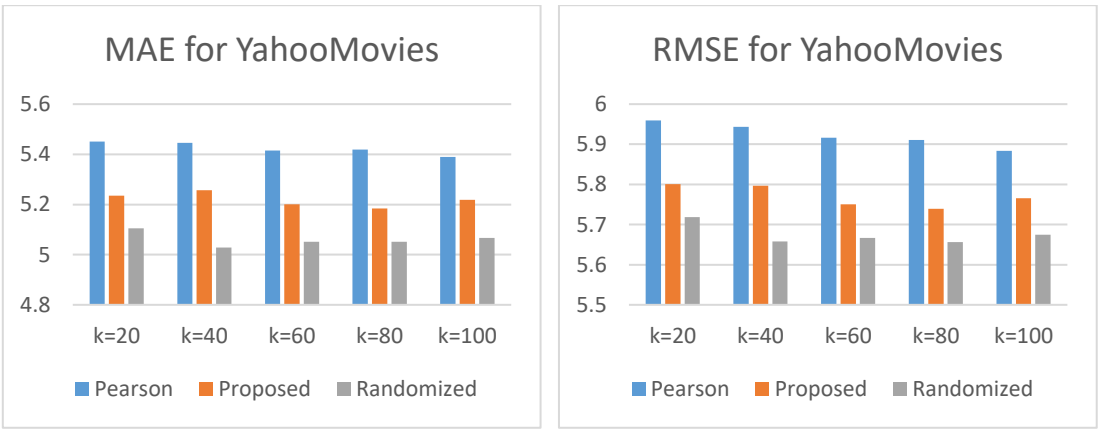
(a) (b)

Figure 5-2 Accuracy results for MovieLens



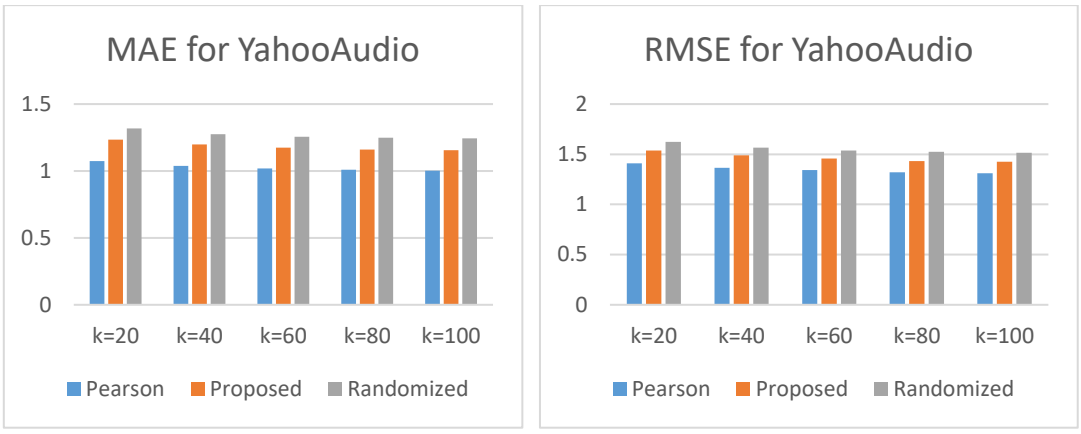
(a) (b)

Figure 5-3 Accuracy results for MovieTweetings



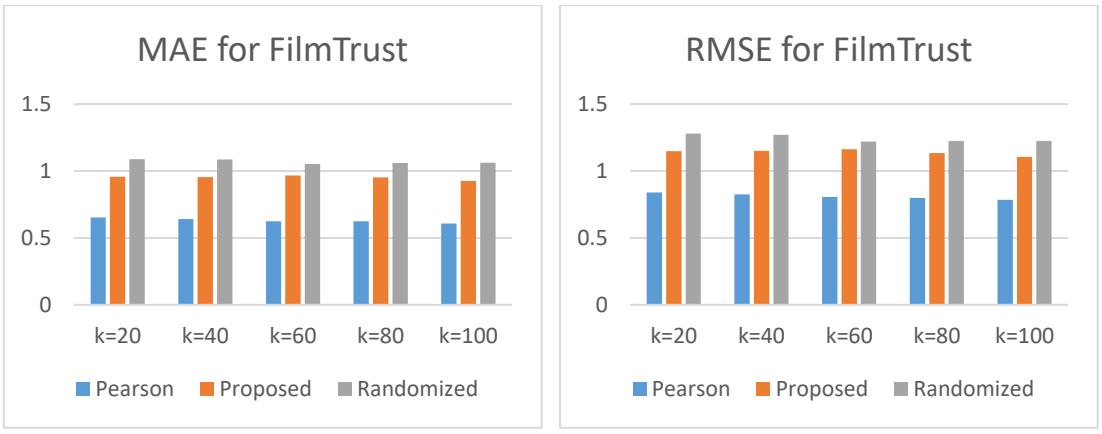
(a) (b)

Figure 5-4 Accuracy results for YahooMovies



(a) (b)

Figure 5-5 Accuracy results for YahooAudio



(a) (b)

Figure 5-6 Accuracy results for FilmTrust

5.4.5 Protection assessment

In evaluating the proposed method for privacy protection, the sum of squared errors (SSE) is used, which is a statistical method that measures the information loss. SSE is used as a measure of distortion on the original data and has also been used as a measure in privacy-reserving collaborative filtering (Casino et al. 2015; Domingo-Ferrer & Mateo-Sanz 2002). SSE is shown in Eq. 5.4. O is the original unaltered rating n *x* m matrix derived from the unaltered dataset with O_{ij} being its elements and P is the n *x* m matrix of the perturbed dataset with P_{ij} being its elements.

$$SSE = \sum_{i=1}^{n}\sum_{j=1}^{m}\left(O_{ij} - P_{ij}\right)^2 \quad (5.4)$$

Table 5.3 shows the results obtained for the datasets. For simplicity reasons the values have been converted to the 10^4 scales.

Perturbation method	**Datasets**				
	MovieLens	**MovieTweetings**	**YahooMovies**	**YahooAudio**	**FilmTrust**
Proposed Method	14	32	43	23	13
Randomized perturbations	17	33	50	15	12

Table 5-3 SSE results

6 Dummy based context privacy

Mobile recommender systems aim to solve the information overload problem by recommending products or services to users of web services on mobile devices, such as smartphones or Tablets, at any given point in time and in any possible location. They utilize recommendation methods, such as collaborative filtering or content-based filtering and use a considerable amount of contextual information to provide relevant recommendations. However due to privacy concerns users are not willing to provide the required personal information that would allow their views to be recorded and make these systems usable. This work is focused on user privacy by providing a method for context privacy-preservation and privacy protection at user interface level. Thus, a set of algorithms that are part of the method have been designed with privacy protection in mind, which is done by using realistic dummy parameter creation. To demonstrate the applicability of the method, a relevant context-aware dataset has been used to run performance and usability tests. The proposed method has been experimentally evaluated using performance and usability evaluation tests and is shown that with a small decrease in terms of performance user privacy can be protected.

6.1 Introduction

The evolution of the concept of e-democracy and its related electronic government services has led to information overload and privacy issues (Drogkaris et al. 2013). Moreover, the information overload problem encountered in numerous online systems / services such as e-commerce and e-government, among others, has given rise to the use of recommender systems (Lu et al. 2015). Such systems have swiftly become necessary to the wider public but at the same time have contributed heavily to an increase in privacy concerns amongst service users. Recommender systems are algorithms and computer software that are designed to provide suggestions for products or services that could be of interest to a user of a website or an online application (Lu et al. 2015; Konstan & Riedl 2012). They are a subset of information retrieval systems whose job is to provide personalized recommendations to users and solve the information overload problem found in various online environments. Recommender systems are valuable to users that do not have the experience or the time to cope with the process of decision making while using the web, particularly where a choice of products or services is available. Recent advances in the field of mobile computing and the rapid evolution of mobile devices such as smartphones and Tablets, has led to the need for advances in the field of mobile recommender systems (Cao, Y., Lu, Y., Gupta, S., & Yang 2014; Ahluwalia et al. 2014; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Ricci 2010; Rodríguez-Hernández & Ilarri 2016). The access to a mobile recommender system at any given point in time and location is called ubiquity, thus an alternative term used to describe such systems is that of ubiquitous recommender systems (Mettouris & Papadopoulos 2014). Additionally, the use of location data and the use of other contextual information, such as the time, weather

information, physical conditions, social and others, have become common in mobile recommender systems (Mettouris & Papadopoulos 2014; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Liu et al. 2013).

These new uses of data have contributed to the creation of more personalized recommendations in mobile environments. It should be noted though that it is not clear whether a specific research domain of mobile recommender systems exists and that only specific goals are set for mobile recommendations, where a mobile application or mobile website is designed and developed for a specific scenario (Jannach et al. 2010; Polatidis et al. 2015; Ricci 2010). In this context different application domains exist, such as those for mobile commerce and tourism related services. Applications designed for any mobile recommendation domain share some common characteristics such as (Polatidis et al. 2015; Rodríguez-Hernández & Ilarri 2016; Gavalas et al. 2014; Liu et al. 2013): All run on a mobile device, such as a smartphone or a Tablet, all provide some form of recommendation, all utilize some form of context and all rely on a wireless connection that could probably be slow.

Mobile context-aware recommender systems heavily rely on context to provide accurate and personalized recommendations in mobile environments. However typical privacy protection techniques such as the use of pseudonyms or the use of anonymity cannot be applied properly since recommender systems rely on the use of personal contextual data. In such a context (Kido et al. 2005) proposed an approach to anonymous communication for location-based services that is based on use of dummies. Similar methods that have been used for privacy protection in location-based services include query enlargement techniques, progressive retrieval techniques and transformation based techniques (Jensen et al. 2009). These are different protection methods that can be adjusted to the context privacy problem found in mobile recommender systems. Privacy is an important part of context-aware mobile recommender systems that has not been properly exploited yet and, to the best of the authors' knowledge, this is the first effort found in the literature to do that. Furthermore, the constant growth of available wireless technologies gives the ability to users to be connected to the Internet from virtually any place and at any given time. Thus, privacy concerns become higher when users want to submit a rating or retrieve recommendations and are in a public place (or, in a different situation, located somewhere private but with friends or family near them). It must be noted that many users are both busy and unsatisfactorily proficient technically, to watch out for themselves. Consequently, thinking in advance about privacy can help both designers and users (Camp 2016). Figure 6.1 shows a typical mobile recommender system. In this scenario, the context is acquired first from mobile device sensors and/or third parties (providing users' profiles and ratings) and then a recommendation method is used to provide recommendations.

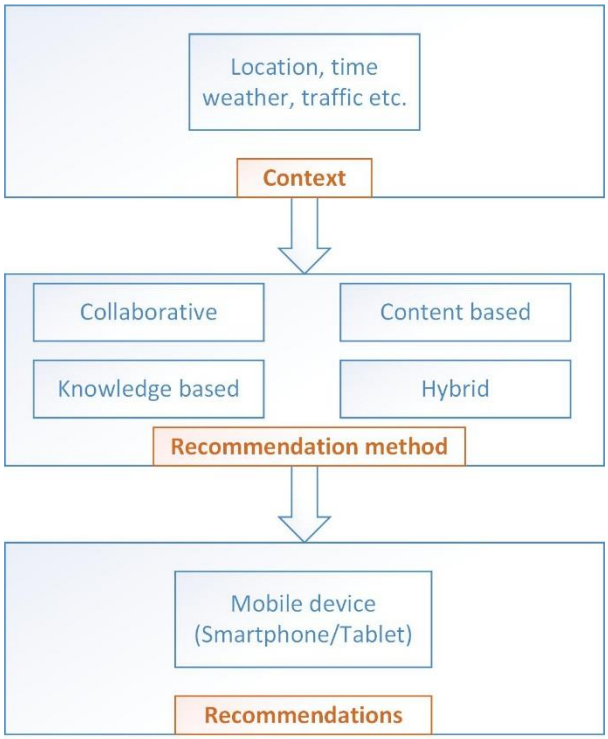


Figure 6-1 A typical mobile recommender system

To protect the user privacy at the context-level the following contributions have been made:

- The author proposed a method that aims to protect the user privacy in mobile context-aware recommender systems.
- It introduces an approach for privacy-preserving context-aware mobile recommendations that is based on realistic dummy context parameter creation.
- Developed a novel privacy-friendly user interface for mobile context-aware recommender systems.
- Experimentally evaluated the method, showing that at the expense of a small performance decrease user privacy can be fully protected.

6.2 Related work

Many works exist that demonstrate the importance of privacy issues in different parts of the recommendations process (from the client part to the server part) in various domains. Mobile recommender systems are used in a variety of different domains, such as the one presented in (Anacleto et al. 2014), where a mobile application is used to provide personalized sightseeing tours to its users. Another mobile application that is designed to provide context-aware tourism information to its user has been provided in (Noguera, Barranco, Segura, & Martinez, 2012).

(Colombo-Mendoza et al. 2015) proposed a context-aware, knowledge-based, mobile recommendation system for movie show times. Privacy is indeed an essential part of mobile recommender systems (Rodríguez-Hernández & Ilarri 2016; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017). However, there is a gap between recommender systems and mobile computing. For example, most of the related work is about the protection of personal user data, such as the ratings. Furthermore, protecting the user location is considered an important aspect in mobile computing services. Thus, in the proposed method the author aims to protect user privacy when contextual parameters are used to provide personalized recommendations in mobile environments. Moreover, the proposed method also protects the privacy at the user interface level. Most of the privacy-protection methods for location-based services perform well for protecting the location of a user when only non-personalized services are requested, the proposed approach delivers recommendations in mobile environments without losing any accuracy and at the same time preserving the privacy of the user at the context and interface level. An example of preserving privacy in collaborative filtering is the use of distribution techniques that use an obfuscation scheme and a randomized dissemination protocol (Boutet, A., Frey, D., Guerraoui, R., Jégou, A., & Kermarrec 2016). Also, the use of ratings perturbations is a well-known approach that is used in collaborative filtering for personal data protection (Polat & Du 2005). Additionally, other privacy protection approaches exist in recommender systems such as (Aimeur et al. 2008) where the use of a semi-trusted third party is proposed. The data are split between the server and the semi-trusted third party, thus no single entity can derive sensitive information and the system can work only if these two separate parties collaborate.

Moreover, approaches in location based services have been developed to protect the location of the user but no other context parameters and this is done for non-personalized services. For example, in (Kido et al. 2005) a technique that is used for location privacy based on dummies is described. Similarly, in (Lu & Jensen 2008) an approach to location privacy is proposed and generates dummy locations based on a virtual grid or a circle. The approach requires only a lightweight front-end that can work tightly in a client-server model. Furthermore, in (Kato et al. 2012) a dummy generation algorithm is proposed to protect location privacy. In this approach, various restrictions are taking place and assume that users do not stop regularly. In (Niu et al. 2014) two dummy based solutions are proposed to achieve *k*-anonymity for privacy-aware users. Also, in (Tran et al. 2010) a binomial mix-based solution is proposed which aims to protect privacy by using a centralized dummy generation mechanism that exploits the activities of each user to perform better in overall. In (Jensen et al. 2009) several different techniques are described for location privacy and include the use of query enlargement, which enlarge the position of the user to a larger set of positions and then send it to the service provider. Additionally, the use of *k*-anonymous approaches is irrelevant

in our case since these types of privacy-protection methods are dependent on the distribution of other users of the system. Besides the above, (Pallapa et al. 2014) is an approach to privacy preservation using adaptive and context-aware user interactions, although it is used in smart environments it is still important. Furthermore, the use of progressive retrieval techniques has been described by the same authors where the client iteratively retrieves results from the service provider without revealing its exact location. Also, the use of transformation based techniques is described in (Kido et al. 2005) that use cryptographic transformation, hence the service provider is not able to identify the exact location and only the client has the decryption functionality to derive the actual results.

In addition, the privacy at the user interface level is a very important aspect in mobile environments. Privacy is certainly an essential part at the Human-Computer Interaction (HCI) level and in more particular at the user interface (Ackerman & Mainwaring 2005; Iachello & Hong 2007). Users of mobile recommender systems suffer from privacy concerns at different levels, including the user interface among others. A shoulder adversary from humans is a factor that needs to be taken into consideration. (Kwon, T., Shin, S., & Na 2014) proposed an approach for privacy protection from human adversaries at the interface level. This work has influenced the proposed approach. Moreover, another stimulus came by (Gamecho et al. 2015): authors propose a method for the automatic generation of accessible user interfaces (although this is not a privacy aware method).

Existing approaches although sufficient in their domain, are only concentrated in one area (such as rating protection or location protection). T motivation is to provide a unified method for privacy protection both at the context level and at the user interface level. In addition, the focus is to serve mobile users. Thus, the author associates the mobile user interface functionality with the context variables. Therefore, the proposed method is focused on privacy protection of users utilizing mobile context-aware recommender systems, by shielding not only the location but also other context parameters and by protecting the mobile user by human adversaries. The detailed explanation of the method can be found in the following section.

6.3 Proposed method

Privacy is becoming increasingly important in mobile computing environments, including the field of recommender systems in such domains. However, efforts have been made towards the location-based services problem, which is only one of the many parameters of context that can be found in context-aware mobile recommender systems. A method that protects the user privacy when context parameters are used for mobile recommendations is proposed. Mobile recommender systems are based on a regular recommendation algorithm such as collaborative filtering, content based

filtering or a hybrid approach. Furthermore, a context filtering method needs to be applied to sort the recommendations and propose the ones that are more relevant according to the contextual parameters. In the method, the author uses collaborative filtering with contextual post filtering while an overview of the recommendation process taking place at the server and shown in Figure 6.2 (Polatidis, Georgiadis, Pimenidis & Stiakakis 2017).

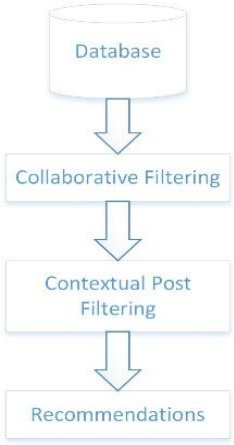


Figure 6-2 Recommendation process service provider overview

6.3.1 Privacy at the context level

To protect the privacy of users requesting context aware recommendations it is necessary to explain the architecture of the system and how it works efficiently.

Mobile device: A mobile device could be a smartphone, Tablet or another device that is porTable and capable of utilizing location through the Global Positioning System or through a wireless network.

Secure communication: It is assumed that a secure communication link is available always between the client (mobile device) and the server (service provider).

Service provider: The service provider, or server, provides personalized recommendations to registered users of mobile devices.

A hybrid client-server recommendation approach is proposed to protect the privacy of the user and provide recommendations within a reasonable time. A mobile user submits a request to the server for recommendations. The ratings of all users are stored at the server therefore collaborative filtering takes place at the server side. Furthermore, when the user makes a request both real and dummy contextual parameters are being sent, such as location, day type, weather, mood, physical and/or others to the service provider. For the recommendation approach to work algorithm 6.1 is called at the mobile device which shows a request submitted from a user to the server. The request includes a dummy creation for every context parameter. The next step is for the service provider

to reply using algorithm 6.2, which is the recommendation process that takes place at the server in typical client-server architectures. Consequently, the real and fake recommendations are being sent back to the mobile device to be sorted out and the real recommendations shown to the user. Figure 6.3 is the interaction between a user of a mobile device requesting recommendations and the service provider. The steps are as follows:

1. Initially, the client-side algorithm 6.1 checks whether a previous generation of dummy data has taken place (within a specified time frame). Then, either it uses these dummy values again, or it randomly generates new dummy values.
2. The next step for the mobile device is to make a request for recommendations to the server, accompanied by a set of real and dummy context parameters' values.
3. The recommendation process takes place at the server, including collaborative filtering and contextual post filtering.
4. The server sends the real and fake recommendations to the mobile device.
5. The mobile device deletes the fake recommendations (the ones based on dummy context values) and presents the real recommendations to the user.

- Note: After the (client-side) creation of the real and dummy context values and just before these are sent to the server (along with the recommendation request), they should be assigned some form of identification: the mobile client should be able to distinguish the real recommendations (those related with real context values) from the fake ones.

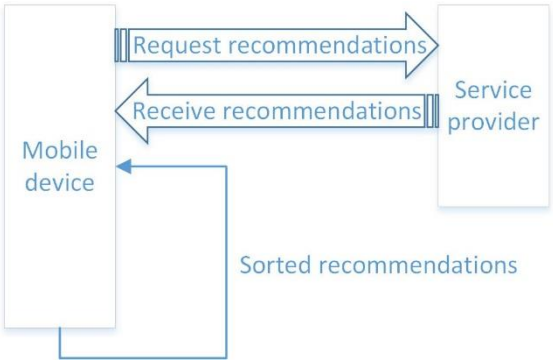


Figure 6-3 Client-Server interaction

Algorithm 6.1: Recommendation request (Mobile client)

Input: User id, Context parameters
Output: Recommendations */* List of recommendations */*

```
Retrieve Location, Context_Parameters[n]
Retrieve Previous_request Location, Context_Paremeters[n]
Retrieve Current_time _time_of_previous_request
If Current_time Within Time Interval with previous_time_request
/* Time intervals could be for example 1 set to morning, 2 set to noon, 3 set to evening
 or any other value set */
Then
        Use Previous_request Dummy_Location, Dummy_Context_Parameters[n]
        /* Dummy_Location and Dummy_Context_Parameters[n] variables contain the dummy
        values that will be passed to the server. These variables are populated during a previous
        execution of this algorithm */
Else
        Generate Dummy_Location, Dummy_Context_Parameters[n]
        /* When no dummy values are available.
        One fake context parameter for each real context parameter is generated */
Request /* to the service provider with parameters */
        User id, Location, Dummy_Location, Context_Parameters[n],
Dummy_Context_Parameters[n]
/* The service provider receives the request, produces the recommendation as shown in algorithm
2 and provides them back to the client */
Receive Recommendations /* real and fake from the service provider */
For (int i=0; i<Recommendations.size; i++)
        If i.hasParameter (Dummy_Location)
                Delete i;
        Else
                For (int j=1; j<=n; j++)
                        If i.hasParameter (Dummy_Context_Parameters[j]
                                Delete i;
                        End If
                End For
        End If
End For
Return Recommendations
```

Algorithm 6.2: Recommendation process (Service provider)

Input: User id, Context Parameters
Output: Recommendations

```
/* Starts with collaborative filtering */
Load User ratings
Load Similarity measure /* Pearson correlation similarity */
Provide Recommendations
/* Contextual post filtering follows */
For (int i=0; i<Recommendations.size; i++)
  If i.hasParameter != (Location || Dummy_Location)
        Delete i;
  Else
        For (int j=1; j<=n; j++)
              If i.hasParameter != (Context_Parameters[j] || Dummy_Context_Parameters[j])
              /* Context_Parameters contains the real context parameters received as input
        from
              algorithm 1. Dummy_Context_Parameters contains the dummy context
        parameters
               received as input from algorithm 1. */
                    Delete i;
              End If
        End For
  End If
End For
Return Recommendations
```

6.3.2 Privacy at the user interface level

Additionally, the author considers privacy to be an important aspect at the user interface level of mobile recommender systems. Thus, use of two different user interfaces is proposed by the author, a regular and privacy-friendly. The interfaces swap according to the context parameters available. Algorithm 6.3 decides, according to relevant context parameters, if a privacy-friendly user interface will be used. If algorithm 6.3 returns a privacy-friendly interface, then algorithm 6.4 needs to run to derive the rating value from that interface. In the case that a regular interface is selected then algorithm 4 is not relevant and won't have to be executed.

Algorithm 6.3: Privacy Decision
/ This algorithm makes a decision if a privacy-friendly interface is necessary, according to the context, or a regular interface */*
Input: location, social
Output: Interface /* either the regular or the privacy-friendly one */

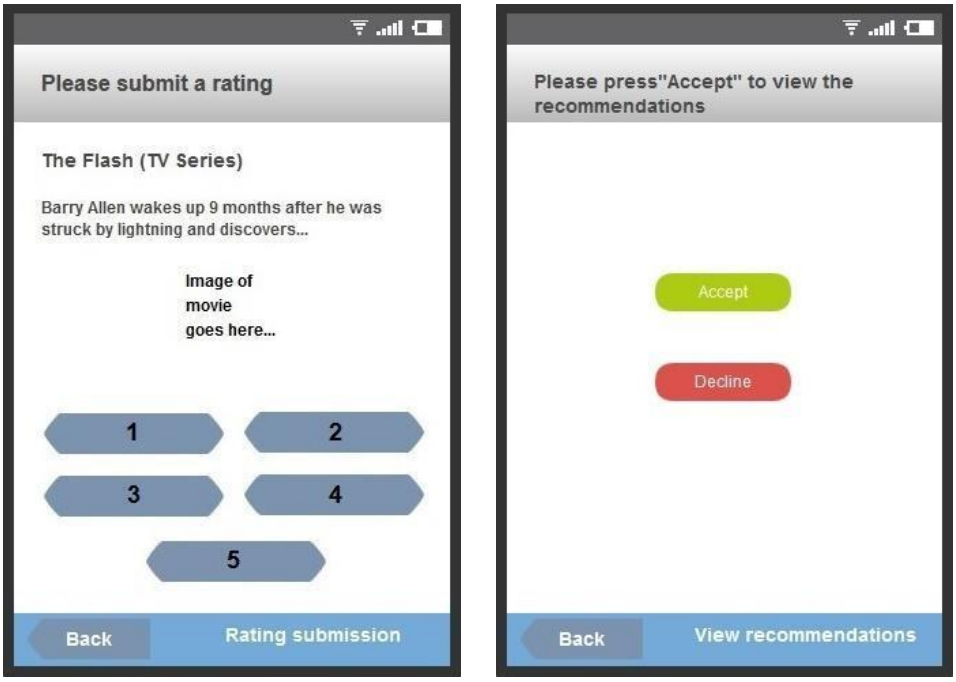
```
If location == home && social == alone
        Then Interface = Regular interface
Else
        Interface = Privacy-friendly interface
Return Interface
```

Algorithm 6.4: Rating Decision
/ This algorithm derives the exact numerical value from the privacy-friendly rating interface */*
Input: Button_number, Button_number.part /* Loads button number eg. 1, 2, 3, 4, 5 and the part of the button eg. left or right */
Output: Rating

```
Switch (Button_number)
        Case 1: If Button_number.part == left    Then Rating = 1
                  Else    /*    Button_number.part == right    */
                        Rating = 2
                Break;
        Case 2: If Button_number.part == left    Then Rating = 2
                  Else    /*    Button_number.part == right    */
                        Rating = 3
                Break;
        Case 3: If Button_number.part == left    Then Rating = 3
                  Else    /*    Button_number.part == right    */
                        Rating = 4
                Break;
        Case 4: If Button_number.part == left    Then Rating = 4
                  Else    /*    Button_number.part == right    */
                        Rating = 5
                Break;
        Case 5: If Button_number.part == left    Then Rating = 5
                  Else    /*    Button_number.part == right    */
                        Rating = 1
                Break;
Return Rating
```

6.3.3 Prototype implementation

A privacy-friendly prototype user interface as shown in Figure 6.4 has been developed. Figure 6.4 (a) is the privacy-friendly rating user interface. Figure 6.4 (b) is a typical warning given to the user if he is in a public location or with someone else before the recommendation list is released.



(a) (b)

Figure 6-4 A privacy-friendly rating interface and a privacy-friendly warning interface

6.4 Experimental evaluation

For the experimental evaluation, a Pentium i3 2.13 GHz with 4GBs of RAM, running windows 8.1 was used. All the algorithms have been implemented in Java and used Collaborative filtering with contextual post filtering. Moreover, a mobile smartphone running android 5.0 was used.

6.4.1 Dataset

For the evaluation part the LDOS-CoMoDa context aware dataset (Košir, A., Odic, A., Kunaver, M., Tkalcic, M., & Tasic 2011) has been used. This is a real dataset that apart from the usual user-rating scale from 1-5 for movies that contains 95 users, 961 movies and 1661 ratings. It also contains 12 contextual variables. Furthermore, the statistical description of the dataset can be found in appendix A.

6.4.2 Context privacy performance evaluation

User Bob is located at his home, which according to the description of the context parameters of the dataset is set to number 1. Moreover, the time of the day is set to number 3 because it is evening time. The other available contextual parameters are social that is set to 1 (alone) and mood which is set to 1 (positive). Now, Bob wants to use his mobile application to recommend him a movie to watch, while he is at home. The following steps take place:

1. Bob starts the mobile application and makes the request.
2. The mobile application automatically selects the current location and the algorithm randomly assigns another location. In this case locations 1 and 2 have been selected.
3. The time is not necessary to be protected. Therefore, the value remains to 3.
4. The social parameter is set to 1 and 3.
5. The mood parameter is set to 1 and 2.

All the information is being sent to the service provider, which then provides movie recommendations according to ratings supplied by the users of the systems and with the use of collaborative filtering and by taking into consideration all the above contextual parameters described in steps 2 to 5. Figure 6.5 shows the performance comparison when the service provider uses one contextual parameter for each type of context and when the second, dummy, parameter is introduced for every type of context. The number of the requested recommendations is set to 5, 10 and 20. It is assumed that user with id no 23 in the dataset is Bob and that's how the experiment took place. It should be noted though that the collaborative filtering method returned 14 relevant results with the provided settings and when the system requested 20. In all cases, all the context parameters applied after the recommendations returned from the collaborative filtering algorithm.

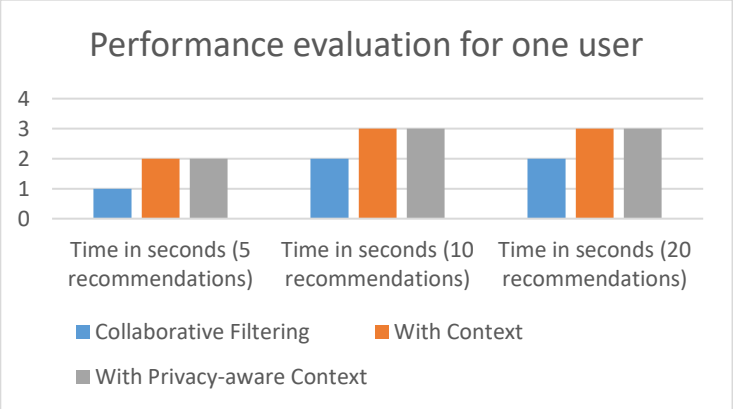


Figure 6-5 Performance evaluation for one user

Figure 6.6 shows the performance results when five users concurrently request for five recommendations each, whereas Figure 6.7 shows the performance results for ten users requesting five recommendations each.

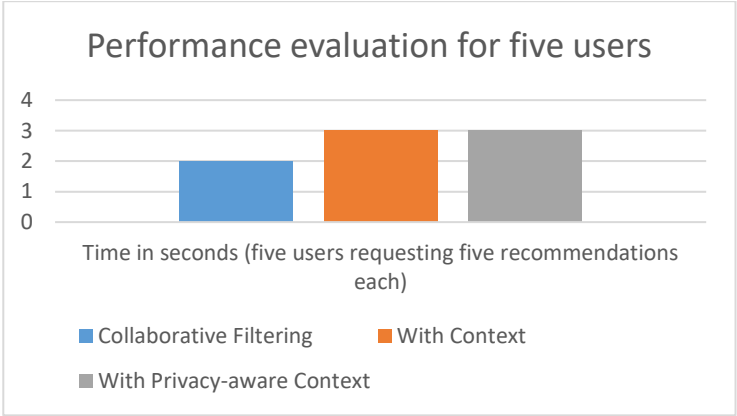


Figure 6-6 Performance evaluation for five users

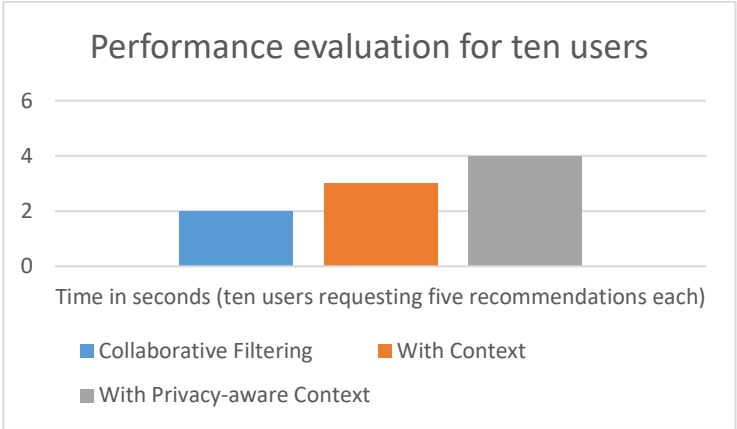


Figure 6-7 Performance evaluation for ten users

Furthermore, a performance regarding the transfer time is necessary and is shown in Table 6.1. Assume that data needs to be transferred from the service provider to the mobile client, a wireless channel is necessary. Supposing that two images for a recommended item are necessary and they are of 100kb each and regular text which has been set to 100Kb. Also, note that rendering time and presentation in the mobile device was not included in these tests. Only transfer times between a computer and a mobile device using a wireless network are included. Moreover, these times may vary depending on the number of concurrent requests to the server and any overheads included.

Number of Recommendations	Wireless Speed	Size in Megabytes	Transfer Time in Seconds
5	11 Mbits	1.5	4
10	11 Mbits	3	8
5	11 Mbits	1.5 (+10% overhead)	5
10	11 Mbits	3 (+10% overhead)	8

Table 6-1 Transfer Time between the computer and the mobile device

6.4.3 User interface evaluation

Initially, the system usability scale has been used to evaluate the user interface of the prototype. The number of participants was 15. Furthermore, the System Usability Scale (SUS) (Brooke 1996) is used. This is one of the most widely used approaches in user interface evaluation (Charfi et al., 2015; Braunhofer et al., 2014). Moreover, the average SUS score computed in 500 studies is 68 and thus this number it may be considered as an accepTable baseline (Braunhofer, 2014). The SUS scale detailed explanation can be found in appendix B. Furthermore, the experimental results of the SUS were based on a user study of 15 participants aged from 20 to 50 years, with usable knowledge of information technology. The users were asked to rate using the privacy-friendly interface five different times each one, and then answer the questions.

- The average SUS score obtained is 72, which exceeds the previously mentioned threshold of '68'.

Additionally, performance evaluation results involved the average entry time in seconds for a regular rating interface from 1 to 5 and the privacy-friendly interface introduced here. More specifically the author measured how long it took in seconds for the same 15 users to submit a rating in both cases. It is believed that this measurement is important since a user needs a certain amount of time to think and interpret that a selection button provides a different rating value from the one that is supposed to give in the privacy-friendly interface. Figure 6.8 shows the results.

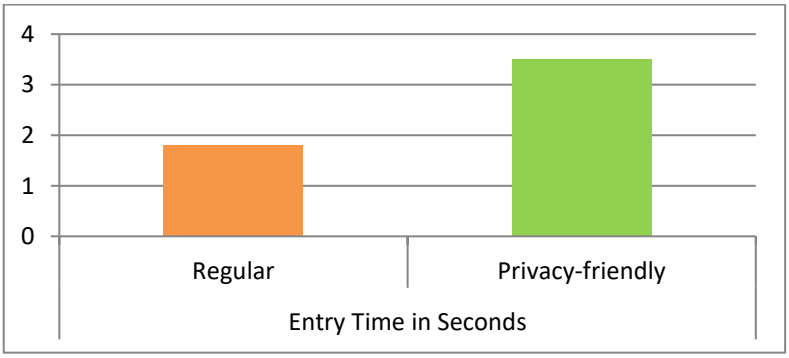


Figure 6-8 Entry Time in Seconds

7 Discussion

In this chapter, the main findings regarding the research questions are summarized and the strengths and limitations of the thesis are discussed.

7.1 Links between mobile and web-based recommender systems

Mobile recommender systems develop on top of e-commerce recommender systems with a specific mobile recommendation scenario in mind (Lathia 2015; Ricci 2010; Polatidis & Georgiadis 2014; Rodríguez-Hernández & Ilarri 2016; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017). These include the following recommendation technologies:

1. A classical recommendation algorithm, such as collaborative filtering.
2. A privacy preserving collaborative filtering method.
3. Utilization of context parameters to provide personalized recommendations to users of mobile devices.

Recommendations provided in mobile devices typically use collaborative filtering as the recommendation algorithm, a privacy preservation method that perturbs user submitted ratings before these are submitted to the server, utilize context parameters and are applied according to a specific recommendation scenario in mind according to the requirements. Therefore, the first research contribution provides two methods that can be used to provide collaborative filtering recommendations of better quality, in which the user will be more interested since they are more highly ranked according to the user rating history. The first method (Polatidis & Georgiadis 2016) is based on a static multi-level approach and the second (Polatidis & Georgiadis 2017) is based on a dynamic multi-level approach. The second research contribution, provides a multi-level privacy-preserving collaborative filtering method (Polatidis, Georgiadis, Pimenidis & Mouratidis 2017) and the third research contribution (Polatidis et al. 2015; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017) provides a method that protects user privacy when context variables are utilized.

7.2 Findings regarding research questions

7.2.1 Research question 1

How can the quality of collaborative filtering recommendations be improved? (Chapters 3 & 4)

Collaborative filtering is widely used in e-commerce to provide suggestions of items or services to users, which include the recommendations of movies, books, general products or users to users (Jannach et al. 2010; Bobadilla et al. 2013; Shi et al. 2014; Polatidis & Georgiadis 2017). Therefore, users can benefit from finding relevant items without the burden of manually searching and

companies could grow extra profit from higher sales of items. Additionally, the computational needs for the company are reduced, since the users normally will not bother making several search attempts to find relevant items. Recommender systems and collaborative filtering are valuable tools for both a user and a company and if an algorithm can produce quality results is beneficial for both sides. Thus, the authors' answer to this question is the proposal of two multi-level methods that add constraints which improve the quality of the recommendations. The proposed methods are based on constraints that can be added either manually or dynamically to the settings of the PCC recommendation method. In the first method, the developer should perform several tests to find the optimal settings for the data and the levels need to be entered as hard constraints on the source code. On the other hand, in the second method this is not necessary, since a series of steps takes place to do this dynamically.

In the experiments, well established evaluation metrics and comparisons to alternative methods have been used throughout (Herlocker et al. 2004; Schröder et al. 2011; a Bellogin et al. 2011; Shani & Gunawardana 2011; Jannach et al. 2010). Initially MAE is used to show that the proposed methods outperform the other recommendation methods, including the state-of-the-art in most cases or having similar results in relatively few cases. However, different ratings scales (e.g. 0-10, 1-5 etc.) produce different accuracy numerical scales, thus making the comparison across different datasets harder to interpret. While accuracy is considered an important aspect when evaluating recommender systems, alternative evaluation methods can supplement these results to support an approach. Further experimental results have been delivered based on the Precision, Recall and F1 metrics and show that the proposed methods perform well both in terms of quality of the Top-N recommendations. The proposed methods produce similar or better results in comparisons against the state-of-the-art recommendation methods both in terms of accuracy and quality (Polatidis & Georgiadis 2016; Polatidis & Georgiadis 2017).

7.2.2 Research question 2

How can user privacy be protected when collaborative filtering systems are used? (Chapter 5)

The proposed privacy-preserving collaborative filtering recommendation method is an important step for developing recommender systems adopted by users with privacy concerns. The author considers the interesting part of the proposal the use of random perturbations within different multiple levels. For example, when using plain randomized perturbations, a random value is added to the rating and is usually derived from a distribution. The author extends this approach by introducing multiple-level privacy protection based on random perturbations, where initially a level

is created and then the random value that will be added or subtracted from the rating is generated randomly with value within the level.

For the experimental evaluation, the accuracy metrics MAE and RMSE have been used along with the SSE and comparisons to other methods (Herlocker et al. 2004; Schröder et al. 2011; A. Bellogin et al. 2011; Shani & Gunawardana 2011; Jannach et al. 2010; Casino et al. 2015). Extensive experiments took place with the results showing that after the proposed perturbation method takes place the accuracy level is still usable and as the neighborhood grows the difference is similar with the non-perturbed data. Furthermore, when using alternative experimentation settings is shown that the difference in accuracy between the unaltered datasets and the perturbed datasets remain very much alike as the previously used settings. Thus, with a small decrease in accuracy the privacy of the users is protected and still accurate recommendations can be provided (Berkovsky et al. 2012; Polat & Du 2005; Casino et al. 2015). Additionally, when comparing the proposed method with an alternative and with similar perturbation settings, it is shown that for every dataset when the ratings are perturbed with each of these methods the results are quite close. Although, if the perturbation range changes, then the output could be different, resulting in an unbalanced system that could either offer less protection with higher accuracy levels or high protection with lower accuracy levels. Therefore, a balance needs to be maintained and tests need to take place when deciding the range of the values that will be used from the perturbation method to have a usable system that protects privacy. Moreover, along with the accuracy experiments SSE has be used to evaluate the protection offered. In SSE, if two identical datasets are compared then the result returned is zero. Therefore, no protection is provided if a zero value is the output. Consequently, in the results it is shown that the values of the proposed method are distant from zero and that privacy is protected.

In the proposed multi-level privacy preservation method for collaborative filtering recommender systems the primary intention is the introduction of multiple-levels in the perturbation process. The author introduces the concept of multiple levels but there are certain implications that need to be considered before developing a multi-level privacy preservation system which include:

1. The number of levels to use.
2. If the range within each level will be fixed, random or based on a dynamic algorithm.
3. The accuracy is relevant to the levels, the perturbation range and the rating scale.
4. Several experiments need to take place to verify the necessary number of levels and perturbation range, to maintain a reasonable tradeoff between accuracy and privacy.

7.2.3 Research question 3

How can user privacy be protected when context parameters are utilized? (Chapter 6)

Privacy is still an open issue and most privacy protection techniques have been based either on personal data, such as user ratings protection or on the use of third party systems for keeping private the exchange of information. Various privacy-protection methods are evolving, capable to preserve privacy at different parts of the recommendations process. The author considers privacy as one of the most important aspects in mobile recommender systems that needs to be protected at the context and at the user interface levels.

With user moving constantly among other people and using a mobile device their privacy should be respected both from other people and service providers (Gamecho et al. 2015; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Polatidis et al. 2015). The author proposes a practical and effective method that protects user privacy in context-aware mobile recommender systems, without the use of a third party. The proposed method automatically generates a set of realistic dummies that are sent from the client to the server, with the real values hidden between the dummy ones. Then the system, with a small utility cost, provides a set of recommendations to the client without any privacy risks. On completion, the client disregards the dummy recommendations, eliminating any surplus and/or confusing data. Such a system may have wide applicability as it can be used in various real life scenarios. In the case of a user using a mobile recommender application for tourists seeking information on local points of interest, the user may protect her contextual information (such as location, social status etc.) when these are being sent to a central server. Another indicative example is that of a user watching a video in a public place. The user may utilize the privacy-preserving user interface in the mobile device to protect the ratings and preferences against indiscreet looks by onlookers or passersby.

The proposed method has been experimentally evaluated both in terms of performance using a real dataset and in terms of usability of the user interface using the same real dataset and a real set of users (Košir, A., Odic, A., Kunaver, M., Tkalcic, M., & Tasic 2011; Gamecho et al. 2015; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Baltrunas et al. 2012). The results show that with a small decrease in terms of performance user privacy at the context level can be protected using realistic dummies. Furthermore, the privacy-friendly user interface has been evaluated using the system usability scale using real users with the average result being above the predefined threshold that is typically used in evaluations (Baltrunas et al. 2012; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017).

7.3 Strengths and limitations

Recommender systems in mobile environments carry most of the characteristics found in e-commerce recommenders. However, it is noted that the use of algorithm selection or privacy preserving technologies should be selected based on the domain and the scenario (Rodríguez-Hernández & Ilarri 2016; Polatidis, Georgiadis, Pimenidis & Stiakakis 2017; Lathia 2015). Furthermore, any method combination could fit well together but every system has its strengths and limitations. In addition, a balance needs to be maintained to provide accurate recommendations, while user privacy is protected.

Collaborative filtering can be used to provide good recommendations to the average user but the static of dynamic multi-level algorithm can be applied instead to provide recommendations of better quality. Furthermore, improved collaborative filtering methods such as the proposed multi-level approaches can be utilized by online companies to reduce the load on their servers and assist users in decision making by providing recommendations of better quality in various domains such as mobile. The proposed approaches are easy to understand and implement when compared to alternatives, while the output is better in most cases or like the state-of-the-art in relatively few comparisons. On the other hand, the multi-level privacy-preserving collaborative filtering method can be applied to protect the user privacy at the rating level, but this comes with a small accuracy cost. It should also be noted that the decrease in terms of accuracy is smaller than the average perturbation method, while the protection level is similar. Furthermore, the loss using the multi-level is smaller than the gain from the proposed multi-level accuracy improvement methods. Thus, user privacy is protected and the quality of the recommendations is still high. In addition, the utilization of context variables is essential to provide relevant recommendations to users of mobile devices. Thus, the protection of their personal context data is essential and this comes with a small decrease in response time. The proposed context-privacy method performs well in terms of context privacy and in terms of user interface protection. With a non-noticeable performance cost, the use of realistic dummies provides efficient protection of user privacy, while the use of an optional privacy friendly user interface provides an extra layer of protection to users who wish to avoid human adversaries.

However, the set of the components have not been evaluated in a newly developed mobile recommender system, which is something that is beyond the scope of this research. Provided that the research components are combined and tested in a various mobile recommendation scenarios then more useful conclusions could be derived. Furthermore, stronger conclusions can be derived with the use of the components in scenarios where real users are engaged and observational evaluation techniques used to monitor the users using the mobile recommender system.

8 Conclusions

Mobile recommender systems are being used in online domains to provide recommendations of items or services to users of mobile devices. Furthermore, these systems are based on specific technologies that can be applied to the specific scenario. Thus, in this thesis, the author proposed a series of methods that can be used in the development of mobile recommender systems to provide recommendations of better quality, while user privacy is protected. In chapter 3 a multi-level collaborative filtering method that improves the quality of the recommendations is proposed. In chapter 4 the concept of dynamic multi-level collaborative filtering is introduced, which is a recommendation method that improves the recommendations. Both these approaches can be used to provide recommendations of improved accuracy and quality to the user. In chapter 5 an efficient multi-level privacy-preserving collaborative filtering method that preserves user privacy at the rating level, while maintaining the accuracy of the recommendations to a high level is proposed. In chapter 6 a context-privacy preservation method is proposed which is based on the use of dummy variables and while it protects efficient the privacy of the user this comes only to a non-noticeable decrease in response time.

In the future, the author plans to investigate the following research directions:

A framework for the delivery of mobile recommender systems. One future research direction is the delivery of a complete framework for mobile recommender systems, with an aim of bridging the gap of mobile computing and recommender systems. Relevant work in mobile recommender systems points towards a direction where an integrated framework needs to exist. Thus, a part of future work could concentrate on the proposal of a framework for developing mobile recommender systems. The framework will utilize different recommendation methods, use them according to the current scenario and provide different options for privacy preserving recommendations depending on the recommendation algorithm and context settings used. However, this research direction includes the evaluation of the system using real world data, well established metrics and by engaging with real users in every stage of the process. Furthermore, each method of such the framework should be evaluated separately from the others and as an integrated whole at the end.

Resolve the complexity of context data. A second future research direction will have to deal with the complexity of context data. Towards this direction both structured and unstructured data can be collected using different sources. The form of these data tends to be both complex and large, thus it will be difficult to process them. It will be necessary to develop appropriate technologies to deal with the storing, search and retrieval of contextual information and as a result more information will be available, which will result to more accurate recommendations.

Personalized advertisements. A third research directions relies with the concept that current mobile recommender systems are developed with specific requirements in mind (e.g. for specific tourism tasks). However, with the availability of different data from different sources the possibility of understanding users better and provide more personalized advertisements or marketing of certain products or services that are more relevant to both the user and its current scenario could be delivered.

Explanations. A fourth research direction should be directed towards the explanation of the recommendations. Since, explanations are very important in recommender systems and with the special characteristic of mobile devices different ways should be found to provide satisfactory explanations to users.

Privacy. A fifth research direction will be towards context privacy protection. Although initial steps have been made more extensive work will need to take place for protecting the privacy of users utilizing enormous amounts of context data while the speed and the accuracy of the recommender system remains reasonable.

Spam. A sixth research direction will have to be towards spam protection. A big challenge for recommender systems is the protection from shilling attacks. Positive or negative ratings are inserted deliberately to promote certain products and while certain measures exist to prevent those, these do not apply to contextual data. Nowadays, with the popularity of mobile devices increasing and with mobile recommender systems available for different domains, research will need to take place to provide systems that will protect users of context-aware mobile recommender systems from possible attackers.

Bibliography

Ackerman, M.S. & Mainwaring, S.D., 2005. Privacy issues and human-computer interaction. In *Computer*. pp. 19–26. Available at: http://scott.mainzone.com/pubs/05-privacy-issues-and-hci.pdf.

Adomavicius, G. & Tuzhilin, A., 2015. Context-aware recommender systems. In *Recommender Systems Handbook, Second Edition*. pp. 191–226.

Aggarwal, C.C. & Yu, P.S., 2008. A General Survey of Privacy-Preserving Data Mining Models and Algorithms. *Privacypreserving data mining*, pp.11–52. Available at: http://www.springerlink.com/index/u4419h332616un75.pdf%5Cnhttp://books.google.com/books?hl=en&lr=&id=8Vr3PtZ3Y7wC&pgis=1%5Cnhttp://www.springerlink.com/index/r3226543596q5204.pdf.

Ahluwalia, P. et al., 2014. Ubiquitous, mobile, pervasive and wireless information systems: current research and future directions. *International Journal of Mobile Communications*, 12(2), pp.103–141. Available at: http://dx.doi.org/10.1504/IJMC.2014.059738.

Ahn, H.J., 2008. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178(1), pp.37–51.

Aimeur, E. et al., 2008. Alambic: A privacy-preserving recommender system for electronic commerce. *International Journal of Information Security*, 7(5), pp.307–334.

Anacleto, R. et al., 2014. Mobile application to provide personalized sightseeing tours. *Journal of Network and Computer Applications*, 41(1), pp.56–64.

Anand, D. & Bharadwaj, K.K., 2011. Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities. *Expert Systems with Applications*, 38(5), pp.5101–5109.

Baltrunas, L. et al., 2012. Context relevance assessment and exploitation in mobile recommender systems. In *Personal and Ubiquitous Computing*. pp. 507–526.

Baltrunas, L. et al., 2011. InCarMusic: Context-aware music recommendations in a car. In *Lecture Notes in Business Information Processing*. pp. 89–100.

Batet, M. et al., 2012. Turist@: Agent-based personalised recommendation of tourist activities. *Expert Systems with Applications*, 39(8), pp.7319–7329.

Beel, J. et al., 2015. Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries*, (February 2014), pp.1–34.

Beel, J. et al., 2013. Research paper recommender system evaluation: a quantitative literature survey. *RepSys*, 20(April), pp.1–35. Available at: http://dl.acm.org/citation.cfm?id=2532508.2532512.

Bellogin, A., Castells, P. & Cantador, I., 2011. Precision-oriented evaluation of recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems - RecSys '11*. p. 333. Available at: http://dl.acm.org/citation.cfm?doid=2043932.2043996.

Bellogin, a, Castells, P. & Cantador, I., 2011. Precision-oriented evaluation of recommender systems: an algorithmic comparison. *... on Recommender systems*, pp.333–336. Available at: http://dl.acm.org/citation.cfm?id=2043996.

Berkovsky, S., Kuflik, T. & Ricci, F., 2012. The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Systems with Applications*, 39(5), pp.5033–5042.

Bilge, A. & Polat, H., 2013. A scalable privacy-preserving recommendation scheme via bisecting k-means clustering. *Information Processing and Management*, 49(4), pp.912–927.

Bobadilla, J., Ortega, F., Hernando, A. & Arroyo, Á., 2012. A balanced memory-based collaborative filtering similarity measure. *International Journal of Intelligent Systems*, 27(10), pp.939–946.

Bobadilla, J., Ortega, F., Hernando, A. & Bernal, J., 2012. A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems*, 26, pp.225–238.

Bobadilla, J. et al., 2013. Recommender systems survey. *Knowledge-Based Systems*, 46, pp.109–132.

Bobadilla, J., Ortega, F. & Hernando, A., 2012. A collaborative filtering similarity measure based on singularities. *Information Processing and Management*, 48(2), pp.204–217.

Boutet, A., Frey, D., Guerraoui, R., Jégou, A., & Kermarrec, A.M., 2016. Privacy-preserving distributed collaborative filtering. *Computing*, 98(8), pp.827–846.

Burke, R., 2002. Hybrid Recommender Systems : Survey and and Experiments. User Modelling and User-Adapted Interaction. *User Modeling and UserAdapted Interaction*, 12, pp.331–370. Available at: http://www.springerlink.com/index/N881136032U8K111.pdf.

Burke, R., 2007. Hybrid web recommender systems. *The adaptive web*, pp.377–408. Available at: http://link.springer.com/chapter/10.1007/978-3-540-72079-9_12.

Cacheda, F. et al., 2011. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Trans. Web*, 5(1), p.2:1–2:33. Available at: http://doi.acm.org/10.1145/1921591.1921593.

Camp, L.J., 2016. Respecting people and respecting privacy. *Communications of the ACM*, 58(7), pp.27–28.

Canny, J., 2002. Collaborative filtering with privacy. In *Proceedings - IEEE Symposium on Security and Privacy*. pp. 45–57.

Cao, Y., Lu, Y., Gupta, S., & Yang, S., 2014. The effects of differences between e–commerce and m–commerce on the consumers' usage transfer from online to mobile channel. *International Journal of Mobile Communications*, 13(1), pp.51–70.

Casino, F. et al., 2015. A k-anonymous approach to privacy preserving collaborative filtering. *Journal of Computer and System Sciences*, 81(6), pp.1000–1011.

Colombo-Mendoza, L.O. et al., 2015. RecomMetz: A context-aware knowledge-based mobile recommender system for movie showtimes. *Expert Systems with Applications*, 42(3), pp.1202–1222.

Domingo-Ferrer, J. & Mateo-Sanz, J.M., 2002. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Engineering*, 14(1), pp.189–201.

Dooms, S., De Pessemier, T., & Martens, L., 2013. Movietweetings: a movie rating dataset collected from twitter. In *Workshop on Crowdsourcing and human computation for recommender systems, CrowdRec at RecSys*. p. 43.

Drogkaris, P., Gritzalis, S. & Lambrinoudakis, C., 2013. Employing privacy policies and preferences in modern e-government environments. *International Journal of Electronic Governance*, 6(2), pp.101–116.

Ekstrand, M.D., Riedl, J.T. & Konstan, J. a., 2011. Collaborative Filtering Recommender Systems. *Foundations and Trends® in Human-Computer Interaction*, 4(2), pp.175–243. Available at: http://www.grouplens.org/node/475.

Fang, H., Guo, G. & Zhang, J., 2015. Multi-faceted trust and distrust prediction for recommender systems. *Decision Support Systems*, 71, pp.37–47.

Gallego, D.V. & Huecas, G., 2012. Generating Context-aware Recommendations using Banking Data in a Mobile Recommender System. *ICDS 2012 : The Sixth International Conference on Digital Society Generating*, (c), pp.73–78.

Gamecho, B. et al., 2015. Automatic Generation of Tailored Accessible User Interfaces for Ubiquitous Services. *IEEE Transactions on Human-Machine Systems*, 45(5), pp.612–623.

Gan, M. & Jiang, R., 2013. Improving accuracy and diversity of personalized recommendation through power law adjustments of user similarities. *Decision Support Systems*, 55(3), pp.811–821.

Gavalas, D. et al., 2013. A survey on mobile tourism Recommender Systems. In *2013 3rd International Conference on Communications and Information Technology, ICCIT 2013*. pp. 131–135.

Gavalas, D. et al., 2014. Mobile recommender systems in tourism. *Journal of Network and Computer Applications*, 39(1), pp.319–333.

Gavalas, D. & Kenteris, M., 2011. A web-based pervasive recommendation system for mobile tourist guides. *Personal and Ubiquitous Computing*, 15(7), pp.759–770.

Georgiadis, C.K. et al., 2017. A Method for Privacy-preserving Collaborative Filtering Recommendations. *Journal of Universal Computer Science*, 23(2), pp.146–166.

Ghorab, M.R. et al., 2011. Towards multilingual user models for Personalized Multilingual

Information Retrieval. *Proceedings of the First Workshop on Personalised Multilingual Hypertext Retrieval - PMHR '11*, pp.42–49. Available at: http://dl.acm.org/citation.cfm?id=2047403.2047411.

Guo, G., Zhang, J., & Yorke-Smith, N., 2013. A Novel Bayesian Similarity Measure for Recommender Systems. In *23rd International Joint Conference on Artificial Intelligence*. pp. 2619–2625.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J., 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pp. 230–237.

Herlocker, J.L. et al., 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1), pp.5–53. Available at: http://portal.acm.org/citation.cfm?id=963770.963772.

Huang, H. & Gartner, G., 2012. Using Context-Aware Collaborative Filtering for POI Recommendations in Mobile Guides. *Notes*, pp.131–146. Available at: http://www.springerlink.com/index/K5081T30071157U7.pdf.

Hussein, T. et al., 2014. Hybreed: A software framework for developing context-aware hybrid recommender systems. *User Modeling and User-Adapted Interaction*, 24(1–2), pp.121–174.

Iachello, G. & Hong, J., 2007. End-User Privacy in Human-Computer Interaction. *Foundations and Trends® in Human-Computer Interaction*, 1(1), pp.1–137. Available at: http://www.nowpublishers.com/product.aspx?product=HCI&doi=1100000004.

Jamali, M. & Ester, M., 2009. TrustWalker: a random walk model for combining trust-based and item-based recommendation. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.397–406. Available at: http://portal.acm.org/citation.cfm?id=1557067%5Cnhttp://dl.acm.org/citation.cfm?id=1557067.

Jannach, D. et al., 2010. *Recommender systems: An introduction*,

Jannach, D. et al., 2013. What recommenders recommend–an analysis of accuracy, popularity, and sales diversity effects. In *User Modeling, Adaptation, and Personalization*. pp. 1–13. Available at: http://link.springer.com/chapter/10.1007/978-3-642-38844-6_3.

Jeckmans, A.J.P. et al., 2013. Privacy in Recommender Systems. *Social Media Retrieval*, pp.263–281.

Jensen, C.S., Lu, H. & Yiu, M.L., 2009. Location privacy techniques in client-server architectures. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. pp. 31–58.

Jingqi Zhang, Jianming Zhu, N.Z., 2014. An improved privacy-preserving collaborative filtering recommendation algorithm. *Pacific asia journal of the association for information systems*.

Kaleli, C. & Polat, H., 2010. P2P collaborative filtering with privacy. *Turkish Journal of Electrical Engineering and Computer Sciences*, 18(1), pp.101–111.

Kato, R. et al., 2012. A dummy-based anonymization method based on user trajectory with pauses. *GIS'12*, pp.249–258.

Kido, H., Yanagisawa, Y. & Satoh, T., 2005. An anonymous communication technique using dummies for location-based services. In *Proceedings - International Conference on Pervasive Services, ICPS '05*. pp. 88–97.

Kikuchi, H. & Mochizuki, A., 2012. Privacy-Preserving Collaborative Filtering Using Randomized Response. *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, 21(4), pp.671–676.

Kim, K., Ahn, H. & Jeong, S., 2010. Context-aware Recommender Systems using Data Mining Techniques. *World Academy of Science, Engineering and Technology*, pp.357–362.

Kobsa, A., 2007. Privacy-Enhanced Web Personalization. *Communications of the ACM*, 50(8), pp.628–670. Available at: http://portal.acm.org/citation.cfm?id=1768197.1768222.

Konstan, J.A. & Riedl, J., 2012. Recommender systems: From algorithms to user experience. *User Modeling and User-Adapted Interaction*, 22(1–2), pp.101–123.

Košir, A., Odic, A., Kunaver, M., Tkalcic, M., & Tasic, J.F., 2011. Database for contextual

personalization. *Elektrotehniški vestnik*, 78(5), pp.270–276.
Koutrika, G., Bercovitz, B. & Garcia-Molina, H., 2009. FlexRecs: expressing and combining flexible recommendations. *Proceedings of the 35th SIGMOD international conference on Management of data*, pp.745–758. Available at: http://portal.acm.org/citation.cfm?id=1559845.1559923.
Kwon, T., Shin, S., & Na, S., 2014. Covert attentional shoulder surfing: Human adversaries are more powerful than expected. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, 44(6), pp.716–727.
Lam, S.K., Frankowski, D. & Riedl, J., 2006. Do you trust your recommendations? An exploration of security and privacy issues in recommender systems. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. pp. 14–29.
Lathia, N., 2015. The Anatomy of Mobile Location-Based Recommender Systems. In *Recommender Systems Handbook*. pp. 493–510. Available at: http://www.springerlink.com/index/10.1007/978-0-387-85820-3.
Lemos, F.D.A. et al., 2012. Towards a context-aware photo recommender system. In *CEUR Workshop Proceedings*.
Liang, T.-P., 2008. Recommendation systems for decision support: An editorial introduction. *Decision Support Systems*, 45(3), pp.385–386. Available at: http://www.sciencedirect.com/science/article/pii/S0167923607000796.
Lin, J. et al., 2011. A context-aware recommender system for M-commerce applications. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. pp. 217–228.
Lin, Y. et al., 2011. Motivate: Context aware mobile application for activity recommendation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. pp. 210–214.
Liu, D.R. & Liou, C.H., 2011. Mobile commerce product recommendations based on hybrid multiple channels. In *Electronic Commerce Research and Applications*. pp. 94–104.
Liu, H. et al., 2014. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, 56, pp.156–166. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-84892441295&partnerID=40&md5=34c3d6ffc22a3f6e40e0d65a8e2907ff.
Liu, Q. et al., 2013. A Survey of Context-Aware Mobile Recommendations. *International Journal of Information Technology & Decision Making*, 12(1), pp.139–172. Available at: http://www.worldscientific.com/doi/abs/10.1142/S0219622013500077%5Cnhttp://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=85924152&lang=zh-cn&site=ehost-live%5Cnhttp://www.worldscientific.com/doi/pdfplus/10.1142/S0219622013500077.
Lu, H. & Jensen, C.S., 2008. PAD: Privacy-Area Aware, Dummy-Based Location Privacy in Mobile Services. *MobiDE*.
Lu, J. et al., 2013. A web-based personalized business partner recommendation system using fuzzy semantic techniques. *Computational Intelligence*, 29(1), pp.37–69.
Lu, J. et al., 2015. Recommender system application developments: A survey. *Decision Support Systems*, 74, pp.12–32.
Massa, P. & Avesani, P., 2007. Trust-aware recommender systems. *Proceedings of the 2007 ACM conference on Recommender systems RecSys 07*, 20, pp.17–24. Available at: http://portal.acm.org/citation.cfm?doid=1297231.1297235.
Melville, P., Mooney, R.J. & Nagarajan, R., 2002. Content-boosted collaborative filtering for improved recommendations. *"Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)"*, (July), pp.187–192.
Mettouris, C. & Papadopoulos, G.A., 2014. Ubiquitous recommender systems. *Computing*, 96(3), pp.223–257.
Miller, B.N., Konstan, J. a & Riedl, J., 2004. PocketLens : Toward a Personal Recommender System. *ACM Transactions on Information Systems*, 22(3), pp.437–476. Available at: http://portal.acm.org/citation.cfm?id=1010614.1010618.

Mishra, R., Kumar, P. & Bhasker, B., 2015. A web recommendation system considering sequential information. *Decision Support Systems*, 75, pp.1–10.

Moradi, P. & Ahmadian, S., 2015. A reliability-based recommendation method to improve trust-aware recommender systems. *Expert Systems with Applications*, 42(21), pp.7386–7398. Available at: http://www.sciencedirect.com/science/article/pii/S0957417415003553.

Niu, B. et al., 2014. Privacy-area aware dummy generation algorithms for location-based services. In *2014 IEEE International Conference on Communications, ICC 2014*. pp. 957–962.

Noguera, J.M. et al., 2012. A mobile 3D-GIS hybrid recommender system for tourism. *Information Sciences*, 215, pp.37–52.

Ozturk, A. & Polat, H., 2015. From existing trends to future trends in privacy-preserving collaborative filtering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(6), pp.276–291.

Pallapa, G. et al., 2014. Adaptive and context-aware privacy preservation exploiting user interactions in smart environments. In *Pervasive and Mobile Computing*. pp. 232–243.

Parameswaran, R. & Blough, D.M., 2007. Privacy Preserving Collaborative Filtering Using Data Obfuscation. In *Granular Computing, 2007. GRC 2007. IEEE International Conference on*. p. 380.

Polat, H. & Du, W., 2005. Privacy-preserving collaborative filtering. *International Journal of Electronic Commerce*, 9(4), pp.9–35. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-24944580011&partnerID=tZOtx3y1.

Polatidis, N. et al., 2015. A method for privacy-preserving context-aware mobile recommendations. In *Communications in Computer and Information Science*. pp. 62–74.

Polatidis, N., Georgiadis, C.K., Pimenidis, E. & Mouratidis, H., 2017. Privacy-preserving collaborative recommendations based on random perturbations. *Expert Systems with Applications*, 71, pp.18–25.

Polatidis, N., Georgiadis, C.K., Pimenidis, E. & Stiakakis, E., 2017. Privacy-preserving recommendations in context-aware mobile environments. *Information and Computer Security*, 25(1), pp.62–79.

Polatidis, N. & Georgiadis, C., 2015. A ubiquitous recommender system based on collaborative filtering and social networking data. *International Journal of Intelligent Engineering Informatics*, 3(2–3), pp.186–206. Available at: http://www.inderscienceonline.com/doi/abs/10.1504/IJIEI.2015.069895.

Polatidis, N. & Georgiadis, C.K., 2017. A dynamic multi-level collaborative filtering method for improved recommendations. *Computer Standards & Interfaces*, 51, pp.14–21. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0920548916301441.

Polatidis, N. & Georgiadis, C.K., 2016. A multi-level collaborative filtering method that improves recommendations. *Expert Systems with Applications*, 48, pp.100–110.

Polatidis, N. & Georgiadis, C.K., 2014. Factors Influencing the Quality of the User Experience in Ubiquitous Recommender Systems. In *Distributed, Ambient, and Pervasive Interactions SE - 35*. pp. 369–379. Available at: http://dx.doi.org/10.1007/978-3-319-07788-8_35.

Polatidis, N. & Georgiadis, C.K., 2013a. Mobile recommender systems: An overview of technologies and challenges. In *2013 2nd International Conference on Informatics and Applications, ICIA 2013*. pp. 282–287.

Polatidis, N. & Georgiadis, C.K., 2013b. Recommender Systems: The Importance of Personalization on E-business Environments. *International Journal of E-Entrepreneurship and Innovation*, 4(4), pp.32–46. Available at: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/ijeei.2013100103.

Pu, P. & Chen, L., 2011. A User - Centric Evaluation Framework for Recommender Systems. *Proceedings of the 5th ACM conference on Recommender systems - RecSys '11*, pp.157–164.

Puglisi, S. et al., 2015. On Content-Based Recommendation and User Privacy in Social-Tagging Systems. *Computer Standards & Interfaces*, 41, pp.17–27. Available at: http://www.sciencedirect.com/science/article/pii/S0920548915000161%5Cnhttp://linkinghub.elsevier.com/retrieve/pii/S0920548915000161.

Ricci, F., 2010. Mobile Recommender Systems. *Information Technology & Tourism*, 12(3),

pp.205–231.
Rodríguez-Hernández, M.D.C. & Ilarri, S., 2016. Pull-based recommendations in mobile environments. *Computer Standards and Interfaces*, 44, pp.185–204.
Ruotsalo, T. et al., 2013. SMARTMUSEUM: A mobile recommender system for the Web of Data. *Journal of Web Semantics*, 20, pp.50–67.
Schafer, J. et al., 2007. Collaborative Filtering Recommender Systems. *The Adaptive Web*, pp.291–324. Available at: http://www.springerlink.com/content/t87386742n752843.
Schröder, G., Thiele, M. & Lehner, W., 2011. Setting goals and choosing metrics for recommender system evaluations. In *CEUR Workshop Proceedings*. pp. 78–85.
Shani, G. & Gunawardana, A., 2011. Evaluating recommendation systems. *Recommender systems handbook*, pp.257–298. Available at: http://link.springer.com/chapter/10.1007/978-0-387-85820-3_8.
Shi, Y., Larson, M. & Hanjalic, A., 2014. Collaborative Filtering beyond the User-Item Matrix : A Survey of the State of the Art and Future Challenges. *ACM Computing Surveys (CSUR)*, 47(1), pp.1–45.
Shokri, R. et al., 2009. Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. *Proceedings of the third ACM conference on Recommender systems RecSys 09*, p.157. Available at: http://portal.acm.org/citation.cfm?doid=1639714.1639741.
Sivapalan, S. et al., 2014. Recommender systems in e-commerce. In *World Automation Congress Proceedings*. pp. 179–184.
Son, L.H., 2014. HU-FCF: A hybrid user-based fuzzy collaborative filtering method in Recommender Systems. *Expert Systems with Applications*, 41(15), pp.6861–6870.
Sotsenko, A., Jansen, M. & Milrad, M., 2014. Using a rich context model for a news recommender system for mobile users. In *CEUR Workshop Proceedings*. pp. 13–16.
Su, X. & Khoshgoftaar, T.M., 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, 2009(Section 3), pp.1–19.
Sun, H. F., Chen, J. L., Yu, G., Liu, C. C., Peng, Y., Chen, G., & Cheng, B., 2012. JacUOD: A new similarity measurement for collaborative filtering. *Journal of Computer Science and Technology*, 27(6), pp.1252–160.
Toch, E., Wang, Y. & Cranor, L.F., 2012. Personalization and privacy: A survey of privacy risks and remedies in personalization-based systems. *User Modeling and User-Adapted Interaction*, 22(1–2), pp.203–220.
Toledo, R.Y., Mota, Y.C. & Martínez, L., 2015. Correcting noisy ratings in collaborative recommender systems. *Knowledge-Based Systems*, 76, pp.96–108.
Tran, M.T., Echizen, I. & Duong, A.D., 2010. Binomial-mix-based location anonymizer system with global dummy generation to preserve user location privacy in location-based services. In *ARES 2010 - 5th International Conference on Availability, Reliability, and Security*. pp. 580–585.
Tveit, A., 2001. Peer-to-peer based Recommendations for Mobile Commerce. *ACM Mobile Commerce Workshop*, pp.1–4.
Wang, W., Zhang, G. & Lu, J., 2015. Collaborative filtering with entropy-driven user similarity in recommender systems. In *International Journal of Intelligent Systems*. pp. 854–870.
Wang, X., Rosenblum, D. & Wang, Y., 2012. Context-aware mobile music recommendation for daily activities. In *Proceedings of the 20th ACM international conference on Multimedia - MM '12*. p. 99. Available at: http://dl.acm.org/citation.cfm?doid=2393347.2393368.
Woerndl, W. et al., 2011. A model for proactivity in mobile, context-aware recommender systems. *Proceedings of the fifth ACM conference on Recommender systems - RecSys '11*, p.273. Available at: http://dl.acm.org/citation.cfm?doid=2043932.2043981.
Yang, W.S. & Hwang, S.Y., 2013. ITravel: A recommender system in mobile peer-to-peer environment. *Journal of Systems and Software*, 86(1), pp.12–20.
Yu, Z. et al., 2006. Supporting Context-Aware Media Recommendations for Smart Phones. *Pervasive Computing*, 5, pp.68–75.
Zhang, S., Ford, J. & Makedon, F., 2006. A privacy-preserving collaborative filtering scheme with two-way communication. In *Proceedings of the ACM Conference on Electronic Commerce*.

pp. 316–323. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-33748679129&partnerID=tZOtx3y1.

Zhu, H. et al., 2014. Mining Mobile User Preferences for Personalized Context-Aware Recommendation. *ACM Transactions on Intelligent Systems and Technology*, 5(4), pp.1–27. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-84919608533&partnerID=tZOtx3y1.

Zhu, T. et al., 2014. An effective privacy preserving algorithm for neighborhood-based collaborative filtering. *Future Generation Computer Systems*, 36, pp.142–155.

Appendix A – Datasets

A.1 The MovieLens 100,000 dataset

The MovieLens 100,000 data set has been collected by the GroupLens Research Project at the University of Minnesota.

This data set consists of:

1. 100,000 ratings (1-5) from 943 users on 1682 movies.
2. Each user has rated at least 20 movies.

The data was collected through the MovieLens web site during a seven-month period from September 19th, 1997 to April 22nd 1998.

This data has deleted users who had less than 20 ratings.

The MovieLens website can be found at movielens.org

A.2 The MovieLens 1,000,000 dataset

The MovieLens 1,000,000 data set has been collected by the GroupLens Research Project at the University of Minnesota.

This data set consists of:

1. 1,000,209 ratings (1-5) from 6,040 users on 3,900 movies.
2. Each user has rated at least 20 movies.

The data was collected through the MovieLens web site from users who joined MovieLens in 2000.

This data has deleted users who had less than 20 ratings.

The MovieLens website can be found at movielens.org

A.3 The MovieTweetings dataset

MovieTweetings is a dataset consisting of movie ratings that were contained in well-structured tweets on Twitter.

This data set consists of:

1. 431,780 ratings (0-10) from 39,363 users on 22,610 movies.

The data was collected using an algorithm that collected twits about movie ratings from twitter. The MovieTweetings dataset can be found at https://twitter.com/mvtweetings

A.4 The Epinions dataset

The Epinions data sets have been collected by the Epinions.com product recommendation website.

This data set consists of:

1. 664,824 ratings (1-5) from 40,163 users on 139,738 products.
2. A direct trust network with 487,183 links

The data was collected through the Epinions.com website from real users

A.5 The Yahoo movies dataset

The Yahoo! Movies dataset is a movie rating dataset that has been collected by Yahoo! Labs and used under license for academic research.

This data set consists of:

1. 211,231 ratings (1-13) from 7,642 users on 11,915 movies.

A.6 The Yahoo audio dataset

The Yahoo! audio dataset is a song rating dataset that has been collected by Yahoo! Labs and used under license for academic research.

This data set consists of:

1. 311,704 ratings (1-5) from 15,400 users on 1,000 songs.

A.7 The FilmTrust dataset

The FilmTrust data sets have been collected by the FilmTrust website.

This data set consists of:

1. 35487 ratings (0.5-4) from 1,508 users on 2,071 movies.
2. A direct trust network with 1,853 links

The data was collected through the FilmTrust website from real users

A.8 The LDOS-CoMoDa dataset

Context Parameter	Values	Description of values
Time	1 to 4	1=morning, 2=afternoon, 3=evening, 4=night
Daytype	1 to 3	1=working, 2=weekend, 3=holiday
Season	1 to 4	1=spring, 2=summer, 3=autumn, 4=winter
Location	1 to 3	1=home, 2=public, 3=friend's house
Weather	1 to 5	1=sunny, 2=rainy, 3=stormy, 4=snowy, 5=cloudy
Social	1 to 7	1=alone, 2=partner, 3=friends, 4=colleagues, 5=parents, 6=public, 7=family
endEmo	1 to 7	1=sad, 2=happy, 3=scared, 4=surprised, 5=angry, 6=disgusted, 7=neutral
dominantEmo	1 to 7	1=sad, 2=happy, 3=scared, 4=surprised, 5=angry, 6=disgusted, 7=neutral
Mood	1 to 3	1=positive, 2=neutral, 3=negative
Physical	1 to 2	1=healthy, 2=ill
Decision	1 to 2	1=By user, 2=By other
Interaction	1 to 2	1=first, 2=number of int, after first

Description of Context Variables of LDOS-CoMoDa dataset

Description	Value
Users	95
Items	961
Ratings	1665
Average age of users	27
Countries	6
Cities	18
Maximum submitted ratings from one user	220
Minimum submitted ratings from one user	1

Statistical description of LDOS-CoMoDa dataset

Appendix B – The System Usability Scale (SUS)

In SUS, the people who participated in the evaluation are asked to answer the following ten questions by choosing one of the five proposed ratings, that range between strongly agree to strongly disagree:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

The SUS uses the following rating format:

Strongly Disagree **1**	2	3	3	**Strongly Agree** **5**

The scoring of SUS is then calculated by using the following rules:

1. For odd questions, such as Q1, Q3, Q5, Q7 and Q9, subtract one from the response received from the user.
2. For even questions, such as Q2, Q4, Q6, Q8 and Q10, subtract the response received from the user from 5.
3. The two above steps scale all the values from 0 to 6.
4. Then we multiply the sum of the scores by 2.5 to obtain a score between 0 and 100.