DYNAMIC MULTI-CONCEPT USER PROFILE MODELLING IN RESEARCH PAPER RECOMMENDER SYSTEMS

MODHI AL ALSHAIKH

PHD 2018

Abstract

The internet and the digital libraries are major sources of information for researchers, and there is an enormous growth of information on these sources. A large number of research papers are available which leads to the information overload problem and hence finding research papers that are related to users' interests become difficult and time consuming. The field of recommender systems aims to solve the information overload problem by filtering information and providing users with relevant results. Although the current recommender systems provide recommendation services to users, different limitations and challenges have not been adequately addressed in the research paper domain. The work presented in this thesis contributes to the development of models and algorithms to the recommender systems in the research paper domain. The main aim of this thesis is to develop a dynamic multi-concept system that is able to recommend research papers of interest at appropriate times. The first contribution of this thesis is modelling dynamic user profiles that are able to adapt to the changes in multiple user interests and to be compatible with the requirements of advanced ontologies. The second contribution is analysing users' reading behaviour with research papers to develop novel short-term and long-term models that are able to adapt dynamically according to a user's changing behaviour during his/her short and long term goals. These models can effectively learn different users' reading behaviours implicitly without the need for any intervention from the user. The third contribution is predicting user's future interests using a novel collaborative filtering approach without the need for the user ratings. All our proposed models are evaluated using offline evaluations with the BibSonomy dataset that contains actual users' records. Our results show that our models outperform the baselines used for comparisons. Finally, we integrated our models to one unified dynamic hybrid system in order to provide recommendations which most closely represent the users' research interests at particular times. The evaluation results indicate that the dynamic hybrid system that models and integrates multiple user interests and concepts can bring substantial benefits to a recommender system in the research paper domain.

Table of Contents

List of Figures

List of Tables

# Acknowledgments

First of all, I thank Allah (God) for the guidance, strength, and the blessing that he granted me in this life. Without his help neither this thesis nor any undertaking of it would have been possible.

A special acknowledgement to my supervisors Dr. Gulden Uchyigit and Dr. Roger Evans for their encouragement, patience, support, and motivation. Their helpful guidance, advices and comments allowed me to focus my research directions and guide me through my PhD journey at the University of Brighton. Their criticisms, constructive suggestions, patience, and insights during the preparation of this thesis, have been integral to the success of this work. Without their professional guidance and help, this work would not have been achieved.

I would like to express my gratitude and gratefulness to my family for their endless trust, support, encouragement and love. Thanks for the lessons of determination and patience that they have taught me to live through difficult times.

Declaration

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

Signed

Dated *April 2018*

Chapter 1. Introduction

This introductory chapter provides an overview of the research that is reported in this thesis. Section 1.1 presents the challenges that motivated this work. This is followed by section 1.2 which presents the research questions. To investigate the research questions, the research objectives are presented in section 1.3. The contributions of this thesis are illustrated in section 1.4. An overview of the structure of this thesis is presented in section 1.5. Section 1.6 presents a list of publications that were produced based on some of the results of this thesis.

1.1 Motivation

The internet and the digital libraries are major sources of information for researchers, and there is an enormous growth of information on these sources. A large number of research papers are available which leads to the information overload problem and hence finding research papers that are related to users' interests become difficult and time consuming (Jain, 2012). The field of recommender systems research aims to solve the information overload problem by filtering information and providing users with relevant results. In general, recommender systems are software tools and techniques that seek to predict users' preferences and interests (Lu et al., 2015). Today, the recommender systems are a major research trend as well as a popular strategy that is applied in a wide range of application domains. One of the most popular application domains is the movie domain, however, there are other application domains such as research papers, books, news, e-learning, music, television, and e-commerce (Park et al., 2012). The main objective of these systems is to provide personalized recommendations with respect to each user's preferences, needs and interests. Recommender systems are reliant on gathering information from users such as users' personal information, browsing history or previous purchasing, and then storing this information in so called *user profiles* in an attempt to model and learn user interests. These user profiles are then used to recommend users with information which is of interest and thus filtering out information which is not relevant. Although the current recommender systems provide recommendation services to users, different limitations and challenges have not been adequately addressed in the research paper domain. One of the main limitations is that they are not dynamic and compatible with newly advanced ontologies. An ontology in the field of information systems is used to represent a set of concepts within a particular field and links that connect these concepts together (Sánchez et al., 2007). In user profile modelling, a concept is a rich representation of a specific topic which might include the entities and characteristics of this topic (Gauch et al., 2007). Ontologies have further split for their original concepts into sub-concepts (Ye et al., 2010). For example, the 2012 ACM Computing Classification System (CCS) (ACM, 2012) replaces the previous 1998 version of the ACM CCS (the '98 ACM CSS'), which has been used by several recommender systems (e.g. Chandrasekaran et al., 2008; Lakkaraju et al.,2008; Kodakateri et al., 2009). The 98 ACM CCS ontology has a three-level hierarchical set of concepts that contains in total 369 concepts (Kodakateri et al., 2009). However, to reflect the rapidly developing field of computing research the 98 ACM CCS ontology was updated to the 2012 ACM CCS to include the new deeper level concepts. The 2012 ACM CCS ontology has a poly-hierarchical ontology and maintains a six-level hierarchical tree with more than one thousand concepts. While ontologies are growing bigger and bigger, finding the relevant research papers that are related to the users' interests becomes a challenging task for the recommender systems.

Another limitation is that most of the recent recommender systems deal with only one concept (topic) context of user's preferences and build the user profile based on this and further, they do not explore multi-concept user profiling. Similar current recommender techniques from domains other than the research paper domain which may use semi multiple concepts are those which utilise short and long terms modelling techniques. The short-term techniques focus on the recent interests of a user, which require fast updating methods, whereas the long-term techniques focus on the stable interests that stay in the user profile for a longer time than the short-term interests (Gauch et al., 2007). However, these techniques are quite limited and do not deal with dynamic multi-concept user profiling. Multi-concept contexts mean that a user can be interested in more than one concept during his/her long and short term goals.

An example scenario will explain why long-term and short-term multi-concept interests should be modelled and considered by a research paper recommender system. For example, a Ph.D. student may in general interested in data mining and databases, so his/her preferences are divided into two concepts and these could be considered as their long-term preferences. The system should be able to analyse the frequently viewed research papers by the student and categorizes them into the correct concepts. Then, it should explore new research papers that belong to these concepts and recommend them to the student. In some situations, he/she may be more interested in more specific research areas in data mining such as collaborative filtering and association rules, these could be considered as the short-term interests with two sub-concepts for data mining. After some time, the student may be no longer interested in association rules, hence the system should remove the keywords that belong to this topic from the user profile. Moreover, after a long period of time the student is not interested in databases and has another concept of interest, then the system should gradually remove databases keywords from the user profile and add new keywords and assign them to the correct concepts. Therefore, the system should dynamically adapt to this kind of multiple information needs to recommend research papers that belong to the student's preferences. However, most of the current recommender system for the research paper domain (Chandrasekaran et al., 2008; Jomsri et al., 2010; Tang and Zeng, 2012; Lee et al.,2013) rely mostly on a static user profile which suffers from collecting information about users that is outdated and irrelevant to their current interests. Such user profiles restrain the dynamic recommendation process because they use the same user information over time which leads to recommend irrelevant and outdated recommendations to the users. Moreover, there is inefficiency to gradually evolve multiple concepts of user preferences during his/her short and long term goals. The importance of this stems from the need to design automatically adaptable user profiling techniques that should keep track of multiple information that are needed by the user. The current techniques are not appropriate for the real-life rapid evolution of the user profile, where the fast deviating in multiple short-term interests may remain undetected and multiple stable long-term interests may not be changed properly according to the most recent user's preferences. An effective recommender system should be able to distinguish between long-term and short-term user interests. Therefore, there is a need for new user profiling and recommendation techniques that automatically adapt to the diverse and frequently changing interests and preferences during user's short and long term goals in the research paper domain.

Another important challenge that faces current recommender systems in the research paper domain is predicting future interests of the users. In other words, the users can be interested in specific concepts but they do not realize that there are other relevant concepts in their research field that may be significant. For instance, a researcher who is interested in the concept "online advertising" may require time to search until he/she realizes that the concept "web mining" is related to his/her research. As a result, there has been an increasing interest in not just modelling the current user interests, but also discovering the future interests of the user. The main difference between determining the current user interests and discovering the future user interests is that determining the current user interests is based only on the user profile, while discovering the future user interests can be based on similar users' profiles. The former type of user interests is usually modelled by using content-based recommendation techniques, while the latter type of interests is modelled using a collaborative recommender system. The content-based systems that deal with current user interests ignore user future interests as these systems assume that a user would have the same interests in the future. In the content-based models, the similar users' profiles are totally ignored, which result in a limited set of recommendations based on current user preferences that are represented in the system. Therefore, a collaborative filtering model is needed to discover the users' future interests by involving the similar users' profiles during the recommendation process. However, finding the similar users is a complicated task in the research paper domain. For instance, in the movie domain, there are many users who have watched the same movies. Therefore, similar users can be found for most users and hence recommendations can be made effectively. However, the research paper domain suffers from the data sparsity problem, where several new research papers have not been read by any user and further, a new user may read only a few research papers. This leads to an inability to successfully locate similar users and hence leads to the generation of weak recommendations. Therefore, there is a need to develop a novel recommendation model that is able to predict future user interests in the research paper domain.

Integrating a content-based model with a collaborative model to generate a hybrid recommender system also poses another challenge in recommender systems. Most of the user profiling techniques in the documents domains focus on specific and isolated problems. For instance, a system for the news domain might focus on the current long-term user interests (e.g. Oh et al., 2014), but not the short-term interest, or a system for the web pages domain might focus on both long-term and short-term interests (e.g. Gao et al., 2013 and Hawalah and Fasli, 2015), but not the user's future interests. Therefore, such systems have limited recommendation capabilities. It is important to integrate all the types of user interests into one unified dynamic system to recommend items of interest at the right times for the user and to be able to rank the recommendation list according to the user's preferences.

Another challenge in current recommender systems is measuring the ranking performance for multiple user interests. In order to evaluate recommender systems, a wide range of measures have been used. These measures can be classified into three categories based on the feature that is being evaluated: measuring the accuracy of rating predictions, measuring the accuracy of usage predictions, and measuring the accuracy of rankings of items (Shani and Gunawardana, 2011). In the first two types, the main aim of the evaluation is to evaluate the accuracy of rating predictions or the retrieval process. However, these evaluations might not be satisfactory to evaluate a system that aims at providing a ranked list of items where the items that are more relevant to users are placed higher on a list than those that are less important. In this case, the order of items is the main concern of such systems. However, the current ranking measures are not designed to measure the performance for multiple concepts. That is, if a user interested in more than one concept (e.g. data mining and object-oriented languages), they cannot evaluate a ranked list for both concepts. Therefore, there is a need to develop a new ranking metric to measure the ranking performance of a recommender system for user's multiple concepts.

1.2 Research questions

Based on all the previously suggested challenges, the main research question in this thesis can be stated as follows:

How can we model users' preferences in dynamic multi-concept contexts in the research paper domain?

From this main research question, some subsidiary questions were identified in order to answer the main research question. These are as follows:

1. What is an effective dynamic technique to use for the representation of the user profiles and the research paper profiles?
2. How can we dynamically represent users short and long term preferences?
3. How can we build a recommendation model that is able to discover future users' interests?
4. How can we integrate different models to develop one unified system that is able to capture, learn, rank and adapt to diverse user multi-concept interests?

Addressing all of the previous questions satisfactorily is still an open problem in the field of recommender systems. Although some studies attempted to address some of these issues independently for other domains than the research paper domain, no system has been developed to address all of these issues in a dynamic and effective way.

1.3 Research objectives

This research aims to enhance the recommendation services in the research paper domain by proposing novel adaptive models for recommender systems. The aim will be achieved through the fulfilment of the following objectives:

- Modelling dynamic user profiles using a rich ontology to provide better recommendations when a user read a large quantity of research papers and has a large distribution of multiple concepts.
- Analysing users' reading behaviour in the research paper domain. Then, developing content-based novel models that are able to dynamically capture and learn multiple users' interests during their short and long term goals.
- Developing a collaborative model that is able to predict user future interests and avoid the sparsity problem in the research paper domain.

- Integrating and exploiting the content-based models for short-term and long-term interests with the collaborative model to provide a user with a recommendation list that contains the most related research papers to his/her interests at the appropriate time.

1.4 Contributions

The work presented in this thesis contributes to the development of models and algorithms that provide multi-concept contexts recommender system for the research paper domain. In particular, this thesis provides a novel dynamic hybrid system that integrates different types of users' interests. The main contributions are as follows:

- **Modelling recommender system using Dynamic Normalized Tree of Concepts (DNTC).** We developed a novel recommender system using the dynamic normalized tree of concepts model that works with a rich ontology that maintains a deep multilevel hierarchy. To the best of our knowledge, our recommender system is the first recommender system for research papers that uses a deep hierarchal ontology such as the 2012 ACM CCS. Our novel DNTC system is able to provide high average precision when a user read a large quantity of research papers and has a large distribution of multiple concepts.

- **Analysing users' reading behaviour with research papers using real users' records.** We used real users' records from the BibSonomy dataset (Knowledge and Data Engineering Group, 2017) over the years 2015 and 2016 for users in the field of computer and information science. This includes 1,642 users and 43,140 research papers. Our analysis involved automatically searching for patterns of users reading behaviour.

- **Developing a model to discover short-term interests with multiple concepts.** We improved the DNTC system to adapt to the user needs for multiple concepts during his/her short-term goals by using a novel personalized dynamic sliding window (PDSW) technique where the window length is adapted according to the user behaviour.

- **Developing a model to discover long-term interests with multiple concepts.** We improved the DNTC system to adapt to the user's long term goals by determining the user's long-term concepts and then selecting the research papers that represent those concepts. The user's long-term profile is built from the selected research papers.

- **Predicting future interests in a research paper recommender system.** We developed a novel collaborative filtering method that computes the similarity between users according to user profiles which are represented using the dynamic normalized tree of concepts model. Then, a community-centric tree of concepts is generated and used to make recommendations.

- **Evaluating different models in isolation.** In this thesis, each model is evaluated in isolation to test different settings and parameters to find the optimum performance of each model in order to effectively evaluate different aspects related to the proposed models.

- **Integrating different types of interests to one unified dynamic hybrid system.** We developed a novel dynamic hybrid system for the research paper domain that integrates different types of interests namely: current short-term, current long-term and future interests by discovering the right balance and cooperation between all our previous models**.** All these models provide a better understanding of the user's needs in the hybrid system to produce the best-ranked recommendation list of the research papers that meets the user's requirements at the right time.

- **The new ranking measure called M_NDCG.** We modified the Normalized Cumulative Discounted Gain (NDCG) evaluation metric to develop a new ranking measurement to measure the ranking performance of a recommender system for multiple concepts.

1.5 Thesis outline

This thesis is structured in eight main chapters as follows:

Chapter 2: Literature Review

This chapter discusses the related literature to the recommender systems. The information that is presented in this chapter includes: collecting user information; user profile representation techniques; short-term and long-term techniques; content-based recommendation approaches; collaborative filtering recommendation approaches and hybrid recommendation approaches. Then we conclude the chapter by highlighting the gap in knowledge for the recommender systems.

Chapter 3: Evaluation Methodologies and Metrics

This chapter presents the evaluation methodologies in the field of recommender systems. The evaluation methodologies can be divided into three different types: offline evaluations, user studies and online evaluations. We illustrate these evaluation methodologies, then present the evaluation metrics that are used to evaluate the performance of recommender systems. Finally, we describe our evaluation methodology and metrics which we have used.

Chapter 4: Modelling Recommender System Using Dynamic Normalized Tree of Concepts (DNTC)

This chapter presents our dynamic content-based recommender system for the research paper domain. This system consists of three main phases: research papers classification phase, dynamic user profiling phase and recommendation phase. The first phase is responsible for preparing research papers and classifying them. The second phase is responsible for tracking user reading activities for research papers. The research papers that are read by the user are used to build a user profile as a Dynamic Normalized Tree of Concepts (DNTC). The third phase is recommendation phase that uses dynamic tree edit distance technique to recommend a set of research papers to the user that belongs to his/her preferences. Finally, we present the experimental evaluations and discuss the results.

Chapter 5: Novel Short-term and Long-term User Modelling Techniques for a Research Paper Recommender System

This chapter provides novel techniques to model the short-term and long-term user interests. First, we present the analysis of users' reading behaviour of research papers using the BibSonomy dataset. Then, we propose our novel user modelling methods for short-term and long-term interests. The short-term model is based on a novel personalized dynamic sliding window (PDSW) technique where the window length is adapted according to the ratio between the number of concepts and the number of research papers recently read by the user. The contents of these research papers are then used to build the user's short-term profile. The long-term model determines the user's long-term concepts and then selects the research papers that represent those concepts. Finally, both of these methods are evaluated in the evaluation section examining different aspects of these models.

Chapter 6: Predicting Future Interests in a Research Paper Recommender System Using a Community-Centric Tree of Concepts Model

This chapter introduces a novel collaborative filtering method to predict users' future interests in the research paper domain. This novel collaborative filtering model does not depend on users' rating, as existing collaborative filtering methods. Our model computes the similarity between users according to the users' profiles which are represented as a dynamic normalized tree of concepts. Then, a Community-Centric Tree of concepts (CCT) is created. The CCT is used to recommend a set of research papers that may relate to the user's future interests. Further, we present our experimental evaluations and results.

Chapter 7: A Dynamic Hybrid Research Paper Recommender System

This chapter introduces a novel hybrid model to integrate different types of interests namely: current short-term, current long-term and future interests by discovering the right balance and cooperation between all our previous models. Moreover, in this chapter we innovate a new ranking measure to evaluate the ranking performance of a recommender system for multiple concepts. We present a set of experiments that have been conducted and discuss the results.

Chapter 8: Conclusions

This chapter discusses the conclusions and outcomes of this thesis. The research limitations and future work are also deliberated in this chapter.

1.6 Publications

As part of this work, the following publications were produced:

1- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling', *In IEEE Eleventh International Conference on Research Challenges in Information Science (IEEE RCIS 2017)*, 200-210. (This publication is related to chapter 4).

2- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'A Novel Short-term and Long-term User Modelling Technique for a Research Paper Recommender System', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017)*, 255-262. (This publication is related to chapter 5).

3- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'Predicting Future Interests in a Research Paper Recommender System Using a Community Centric Tree of Concepts Model', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017)*, 91-101. (This publication is related to chapter 6).

Chapter 2. Literature Review

The enormous growth of information on the internet makes finding information challenging and time consuming. Recommender systems are software tools and techniques that seek to predict users' preferences and interests (Lu et al., 2015). Recommender systems provide users with suggestions for items a user may wish to utilize. "Item" is the general term used to denote what the system recommends to users, for example, movie or book (Ricci et al., 2011). The suggestions are related to the recommender systems' technologies that study patterns of user's behaviour to know what a user will prefer from among a collection of items he/she has never experienced (Ricci et al., 2011). There has been an increase in the development of recommender systems in diverse application domains. One of most the most popular application domains is movies; however there are other application domains such as music, television, books, news, research papers, e-learning, and e-commerce (Park et al., 2012). In general, recommender systems are composed of two essential components: the user profiling technique and the recommendation method. Figure 2.1 shows a general overview of a recommender system. During the user profiling technique, user data is collected and processed to generate the user's profile. A user's profile represents the information personalized for an individual user from his/her past preferences (Gauch et al., 2007). Then, the user's profile is used to recommend items of interest to the user. A wide range of techniques and approaches have been developed to provide recommendation services in different domains of application. This chapter is organized as follows. First, the difference between recommender systems and search engines is presented in section 2.1. Then, the user profiling techniques are presented in section 2.2. The recommendation approaches are presented in section 2.3. Then, the gap in knowledge in the current presented studies in the literature is discussed in section 2.4.

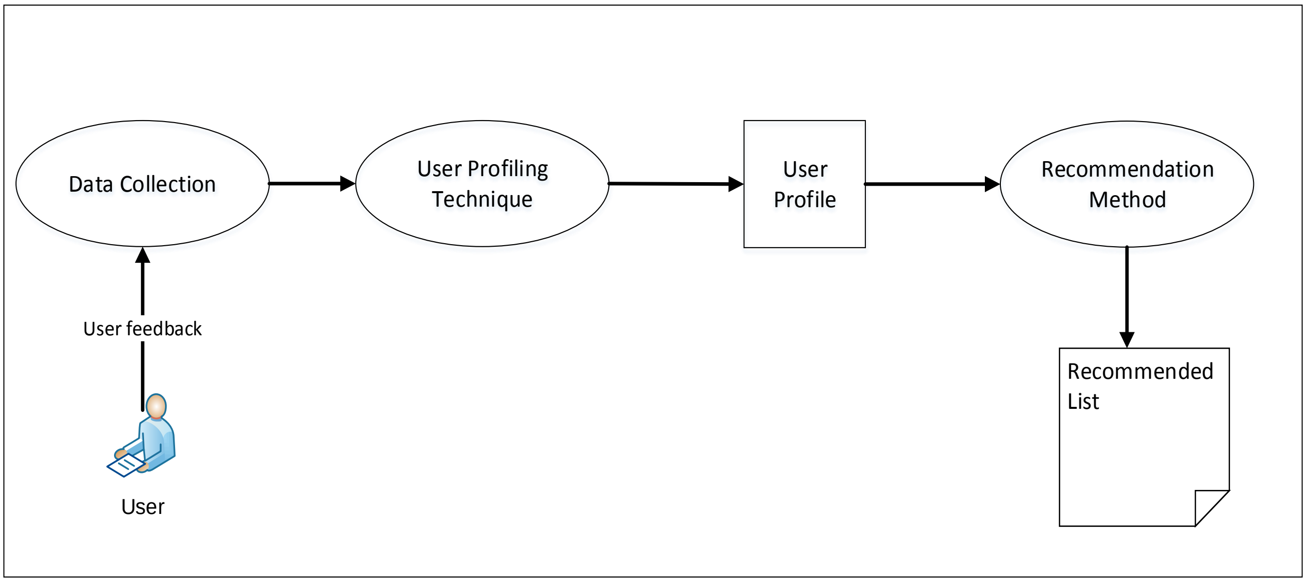

Figure 2.1. A general overview of a recommender system.

2.1 The difference between recommender systems and search engines

The research of recommender systems is relatively new compared to the research of search engines (Ricci et al., 2011). In the beginning of recommender systems development, the researchers split it from search engines and produced two different communities (Jack, 2013). Recommender systems rely on collecting information from users such as user's personal information, browsing history or previous purchasing behaviour, and storing of such information in what it is called the user profile. The user profile is then used to provide the user with relevant information. Search engines do not use a user profile to return results. Nowadays, these two communities are increasingly coming together as advances in personalized search engines include lessons learned from recommender systems' techniques (for example, creating a user profile) and as recommender systems start exploiting well established search engine techniques (for example, learning to rank items) (Jack, 2013). Therefore, it is becoming less relevant to distinguish between personalized search engines and recommender systems based on their underlying technologies. Now their primary difference is in how users interact with them. Search engines require a user to manually enter a search query in order to return a list of relevant results (Kotkov et al., 2016). Here, the user has an idea of what he/she is looking for and the item may or may not exist but if it does then the search engine tries to retrieve it. The user may need to refine his/her query as he/she sees what results are returned and he/she widens or narrows the search. An advanced personalization search system can create and use a user's profile to provide him/her with personalized results when a user starts a new search session (Leung and Lee, 2010). Search engines wait until the user starts a session of search to provide him/her with interesting results. On the other hand, a recommender system can automatically model and learn the user's preferences and recommends items of interest. The user does not use a search query to retrieve results (Kotkov et al., 2016), as in search engines. It may be problematic for the user to choose keywords for the search query especially if they are new to the domain which they are searching. Recommender systems provide a solution to this problem by automatically capturing the user's preferences and recommending items of interest. Moreover, recommender systems recommend interesting items to the user as new items are added to the system. Hence, personalized search engines are not the same as recommender systems. Both can provide personalized services that match user's needs, but the difference is not what they do or the technologies that are used, the main difference is how the user interacts with them (Jack, 2013 and Kotkov et al., 2016).

2.2 User profiling techniques

Data that reflect user's preferences, interests, and goals need to be collected, represented, constructed and exploited in order to provide recommendations. These processes usually are referred to as the process of building and modelling a user profile. Firstly, the user preference related information is collected, then a technique is employed to build the user profile.

2.2.1 Collecting user information

Recommender systems collate information about a user that reflects user's preferences, interests and goals. This information can be used to model the user's profile, which enables the recommender system to provide a personalized recommendation to the user. In general, user related information can be acquired explicitly or implicitly (Challam et al., 2007).

To acquire user information explicitly necessitates the user to provide information to the recommender system, i.e. rating an item (Challam et al., 2007). Such a method is considered to be a straightforward method as the user is explicitly asked to provide information regarding their needs, preferences and interests. For example, in order to obtain user information, a recommender system may ask the user during the registration process to complete a form with a list of predefined topics of interest (Kritikou et al., 2008). In Jomsri et al. (2010) each user is allowed to create his/her keywords when he/she posts a research paper. Netflix (Netflix, 2014) askes the users to rate a set of movies, then the users' ratings are collected (Yue et al., 2014). However, collecting data explicitly has some drawbacks. The main drawback is that users need to provide information about their interests and needs manually. Asking users to fill forms or rate items every time they browse or purchase items is expensive in terms of time and effort.

Implicit user information can be collected by monitoring user's activities and behaviour such as web sites visited and documents read. Collecting and tracking user information implicitly can provide recommender systems with rich and sufficient information about user interests and preferences. For instance, Hawalah and Fasli (2015) designed a system that tracks the user's visited web pages. For each web page, the content, time stamp of the visit and the duration of the visit are observed. Zeb and Fasli (2011) applied implicit collecting method to collect user information and model user profile by tracking user's clicks on news pages. The terms and time stamp are extracted from each visited news page to build the user profile. In Kodakateri et al. (2009) a research paper recommender system is presented. Their system firstly tracks click histories and visited research papers for a user using CiteSeer digital library (CiteSeer, 2008), and then processes the collected information in order to build the user profile, which is then used to provide efficient recommendation services.

Overall, implicitly collecting user information has both advantages and disadvantages. One clear advantage is that a user is not burdened with filling forms or rating items. However, extracting user interests implicitly requires addressing more complex processes and require adopting different techniques to analyse the collected information. Systems such as Amazon (Amazon, 2009) take advantage of both explicit and implicit methods. User information is acquired both by active ratings and implicit behaviour tracking (Kodakateri et al., 2009). Also, Agarwal and Singhal (2014) and Alhabashneh et al. (2015) present user profiling systems that capture user interests both explicitly and implicitly. The explicit information in (Agarwal and Singhal, 2014) is recorded at the time of user registration, whereas implicit information is gathered based on three factors. These factors are: the user's click behaviour, recency of session and active duration in the user session. The implicit information in (Alhabashneh et al., 2015) includes: visit time stamp, reading time, number of mouse clicks, mouse movement, mouse scrolling, bookmark, save and print. The explicit information is collected through asking the users to rate the visited documents. Table 2.1 summarizes the studies in our literature review considering how user information is collected in each study and the application domains.

2.2.2 User profile representation techniques

In this section, we discuss the development of user profile representations techniques. As user interests and preferences are the key elements in any recommender system (Gauch et al., 2007), these interests and preferences can be represented in a user profile in two main categories: weighted keywords and weighted concepts using an ontology. Under these two categories, a number of user profiling techniques have been used in recommender systems. We focus on user profiling for documents recommendation such as news pages, web pages and research papers. Documents contain raw information; therefore they require some pre-processing steps to extract data before using a user profile technique. This raw information can be stored in a raw

Reference	Information collection Type	Source of information	Application domain
Jomsri et al., 2010	Explicit	Allow the user to create his/her keywords	Research papers
Yue et al., 2014	Explicit	Asking the user to rate a set of movies	Movies
Hawalah and Fasli, 2015	Implicit	Tracking web pages and user behavior	Web pages
Zeb and Fasli, 2011	Implicit	Visited news pages by the user	News
Kodakateri et al., 2009	Implicit	Past click histories and visited research papers	Research papers
Lee et al., 2013	Implicit	User’s previously published papers	Research papers
Agarwal et al., 2005	Implicit	Tracking the log of research papers that accessed by the user	Research papers
Fanaee-T and Yazdi, 2011	Implicit	User’s posts in the forum	Online forums
Tang and Zeng, 2012	Implicit	The visited research papers by the user	Research papers
Chandrasekaran et al., 2008	Implicit	User’s previously published papers	Research papers
Oh et al., 2014	Implicit	Tracking the visited news pages	News
Kacem et al., 2014	Implicit	User’s tweets	Twitter
Amazon	Explicit and Implicit	Ratings and behaviour tracking	E-commerce
Agarwal and Singhal, 2014	Explicit and Implicit	Registration form and tracking the visited news pages	News
Alhabashneh et al., 2015	Explicit and Implicit	Asking the user to rate a set of documents and tracking the user behaviour	Documents (Enterprise search)

Table 2.1. An overview of user information collection methods.

document file which is then used by a pre-processing component (Hawalah and Fasli, 2015 and Lee et al., 2013). Figure 2.2 presents some of the pre-processing steps, which are: clean noise, tokenize, remove stop words and stemming. First, all the noise in a document can be removed. For example, if the document is a web page, then HTML (Hyper Text Markup Language) tags can be removed (Hawalah and Fasli, 2015). Then, the document is tokenized to discover all terms. For more efficient processing, the dimensionality of terms is reduced by removing stop words (such as 'and', 'or' and 'the') using a stop list (Hawalah and Fasli, 2015 and Lee et al., 2013). Then, a stemming algorithm, such as Porter stemming (Sparck and Willett, 1997), is applied to return each term to its stem.

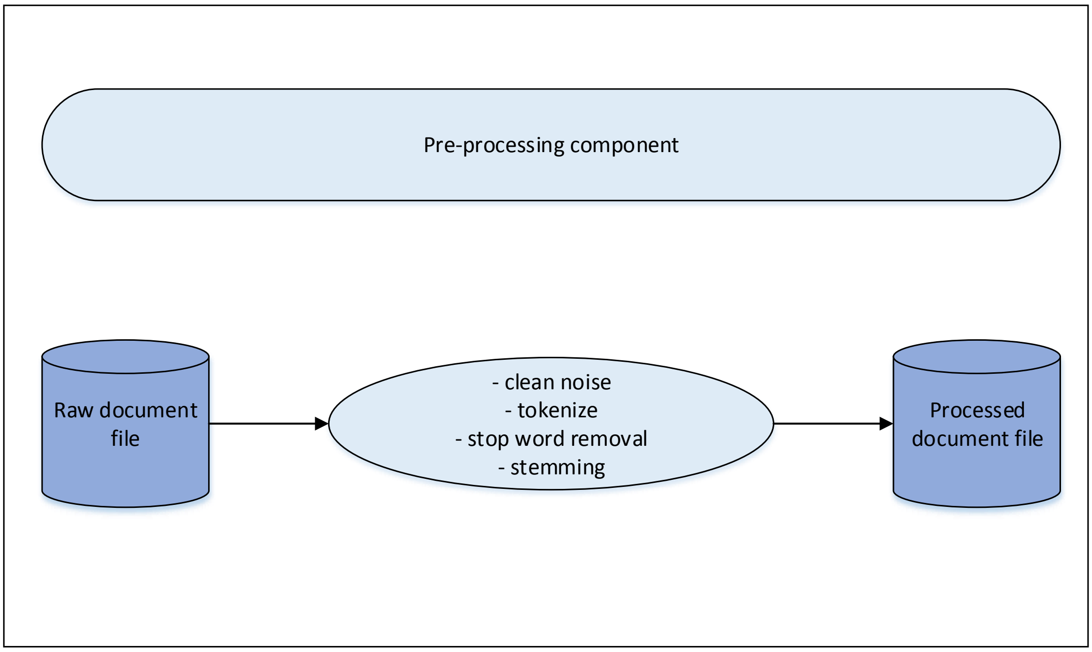

Figure 2.2. Pre-processing component.

2.2.2.1 Weighted keywords

Weighted keywords technique is the most common representation for user profiles (Gauch et al., 2007). The keywords can be automatically extracted from web pages or documents during user browsing, web pages bookmarked or saved by the user, or the keywords are explicitly provided by the user (Gauch et al., 2007). Keywords are associated with weights to create numerical representations of user interests. The weight represents the keyword's importance in the user profile. Different techniques are used to assign weighs to each keyword, the rest of this section presents some of these techniques.

Lee et al. (2013) developed a system that creates keywords by extracting keywords from user's previously published papers, assuming that researchers will be interested in similar research papers to their previous research topics. The information of a research paper is retrieved by implementing a web data gatherer for two research papers engines: IEEE Xplore (IEEE, 2011) and ACM (ACM, 2011) digital libraries. They retrieve the published research papers for each user using these two engines. Each published paper is represented by a set of words that are selected from the title, paper's keywords, and abstract. The data is processed by applying the Bag-of-words model to the corpus. In the Bag-of-words model, each word which appeared in the document corpora represented as an attribute, and then each document is represented by a bit vector, indicating whether each word appears or not. This user profiling technique is very simple because the user profile contains only the words without actual weighting mechanism. Lee et al. (2013) stated that the accuracy of their technique can be improved by using the weighting method Term Frequency-Inverse Document Frequency (TF-IDF) vectors weighting scheme (Dillon, 1983).

The TF-IDF scheme is one of the widely used weighting methods in information retrieval (Challam et al., 2007 and Beel et al., 2016). TF-IDF is a statistical measure used to evaluate how important a keyword is to a document in a collection or corpus (Dillon, 1983). The importance increases proportionally to the number of times a keyword appears in the document but is offset by the frequency of the word in the corpus. TF-IDF is calculated as follows:

$$TF\text{-}IDF\ (t) = TF\ (t) * IDF\ (t) \qquad (2.1)$$

Where *TF (t)* is Term Frequency that measures how frequently a term *t* occurs in a document *d*:

$$TF\ (t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document\ d}{Total\ number\ of\ terms\ in\ a\ document\ d} \qquad (2.2)$$

And *IDF(t)*is Inverse Document Frequency that measures the importance of a term *t* across all documents in the corpus:

$$IDF(t) = \log\left(\frac{Total\ number\ of\ documents\ in\ the\ corpus}{Number\ of\ documents\ with\ term\ t\ in\ the\ corpus}\right) \quad (2.3)$$

After representing each user's profile as keywords vectors, some comparison methods such as cosine formula (Dillon, 1983) can be used to compare a user's profile with the documents in a system's collection. Cosine similarity calculated as follows:

$$CosinSim\ (d_1, d_2) = \frac{\sum_{i=1}^{n}(w_{i1} * w_{i2})}{\sqrt{\sum_{i=1}^{n} w_{i1}^2} * \sqrt{\sum_{i=1}^{n} w_{i2}^2}} \quad (2.4)$$

Where n is the number of the keywords vectors, w_{i1} is the TF-IDF weight for keyword i in document d_1 and w_{i2} is the TF-IDF weight for keyword i in document d_2. For example, Jomsri et al. (2010) propose a framework for a tag-based research paper recommender system. Their approach exploits the use of sets of tags for recommending research papers to each user. They allow each individual user to create his/her keywords as user self-defined tags. Users in this framework post their research papers and the system asks users to create their own tags to attach them to the posted research papers. They consider these tags as keywords in a user profile and the weights for these keywords are assigned by using the TF-IDF weighting scheme. Zeb and Fasli (2011) propose a technique that constructs a probabilistic user profile. The users subscribe through an RSS (Rich Site Summary) news aggregator to create their user profile. The user profile is then built using a probabilistic model based on implicit user feedback (click response) over a period of time. For each visited news page, the terms are extracted as keywords and ordered according to their frequency in the news page. A weight to each term in the user profile is assigned initially based on term's frequency in the visited document set. Then, a probabilistic method is used to estimate the user's interests and recommend new interesting news to the user. Alhabashneh et al. (2015) present an adaptive fuzzy logic based recommender system for enterprise search. The successful user queries that are led to document visits are pre-processed and the query terms are extracted to calculate the TF-IDF matrices for the terms and used by the fuzzy system. The fuzzy system is used to create profiles for the user, search task and document.

Although the weighted keywords approach may be a simple method to build a user profile, it has few critical disadvantages. One limitation is that it is not suitable for a more complex representation of user profiles (Gauch et al., 2007). This is because representing user interests as simple keywords increases the ambiguity as it lacks semantic information. Therefore, it has to capture and represent most of the words to represent user interests, hence it requires a large amount of user feedback and monitoring in order to learn the terminology of user interests. If a system had more knowledge of semantic relationships, it could use a training data more efficiently, and so need less user interaction with the user to build an accurate profile. One way of achieving this is through the use of ontologies. Ontologies are trained on examples for concepts. Therefore, beginning with an existing mapping between words and concepts in a reference ontology can build robust users' profiles with less user feedback and monitoring (Gauch et al., 2007). The following section explains the weighted concepts techniques using an ontology.

2.2.2.2 Weighted concepts using an ontology

The definition of a concept is a "general idea or notion that corresponds to some class of entities and that consists of the characteristic or essential features of the class" (Farlex, 2016, p.1). In user profile modelling, a concept is not just a simple keyword, but it is a rich representation of a particular topic which might include the entities and characteristics of this topic. According to Gauch et al. (2007), a user profile that uses weighted concepts representation consists of a set of nodes that represent conceptual topics and links between these nodes that reflect possible relationships between them. Complex users' profiles can be constructed using a reference ontology. The relationships between concepts in an ontology are explicitly specified and the resulting user profile may include a wide variety of relationship types and richer information (Gauch et al., 2007). The term ontology has its origins in the field of Philosophy, and has been applied differently based on the domain that uses it (Sánchez et al., 2007). An ontology in the field of information systems is used to represent a set of concepts within a particular domain and links that connect these concepts together. A broadly accepted definition of ontology in the context of information systems area was introduced by Gruber (1993, p.2) who states: "an ontology is an explicit specification of a conceptualization". It is usually modelled in a hierarchical way in which parent concepts are linked to child concepts. An ontology provides a clear illustration of the contents of a particular domain of an application and provides a richer representation than flat representations of information in a way that semantic and structural relationships are defined explicitly. It provides a rich representation of concepts as well as a rich variety of relationships among them. Unlike the simple method of user profile representation as weighted keywords, weighted concepts using an ontology provide a more powerful, deeper and broader concept hierarchy representation for user interests (Gauch et al., 2007). A user profile represented as weighted concepts using an ontology has been applied widely in developing recommender systems. Examples of ontologies that are used in recommender systems are: Open Directory Project (ODP) (ODP, 2011) and ACM Computing Classification System (ACM CCS) (ACM, 2012). An ontology is a rich knowledge representation which has been shown to provide a significant improvement in the performance of user profiling models in recommender systems (Challam et al., 2007).

When a recommender system uses an ontology, an additional process is required. This additional process called classification, which classifies each item to the corresponding concept(s) in a reference ontology. There are different classification methods to map a user's preference to the appropriate concept(s). A commonly used technique is the cosine similarity method with TF-IDF weighting scheme (Dillon, 1983). For example, Fanaee-T and Yazdi (2011) classify the research papers by using the TF-IDF vectors weighting scheme to give weight to keywords. Then the Latent semantic indexing (LSI) (Dumais, 1988) technique is employed to discover concepts with similar semantics relations. Another example, Hawalah and Fasli (2015) use the TF-IDF vectors weighting method to discover concepts with similar semantics relations. Hawalah and Fasli (2015) employ cosine similarity method to compare a web page visited by a user with a concept's represented document. Also, Tang and Zeng (2012) compute the TF-IDF values of keywords, then they innovate automatic clustering algorithm to discover the related concept for each research paper visited by the user.

Most research which use an ontology for user profile modelling use it in a similar way to the weighted keywords where the concepts are represented as vectors of weighted features, however, the features represent concepts rather than words

(Gauch et al., 2007). Moreover, there is research (Chandrasekaran et al., 2008) that represents a user's profile as a tree of concepts rather than vectors of concepts. In the following section, we discuss in detail both methods to use weighted concepts, which are: vectors of concepts and tree of concepts.

2.2.2.2.1 Vectors of concepts

Vectors of concepts means that concepts from an ontology are represented as vectors of weighted features, but the features represent concepts rather than keywords (Gauch et al., 2007). Various techniques are applied to express the degree of user interest in each concept. The rest of this section presents some of these techniques.

Agarwal and Singhal (2014) employ OWL (Web Ontology Language) (OWL, 2013) to build a user profile. Their system periodically gathers visited news pages by using unique features of RSS feed news items and arrange them in chronological order. The user profile consists of concepts that are interesting to the user. Concepts will be given weights based on a number of clicks in a session, recency of session and active session duration. Fanaee-T and Yazdi (2011) employ a recommender system on online forums that suggest favourite's topics of users according to their tastes. In their system, there are two steps to build a user profile. Firstly, the number of user posts in the forum will be considered in addition to each word frequency as its weight. For example, if a user participated in a discussion *three* times and the word "Java" exists *five* times in the discussion, then the word "Java" will be allocated with weight *15* in the user profile. Secondly, the user profile is enriched with the Wordnet ontology (Gupta et al., 2002). This ontology is used to add three vectors in the user profile to each existing word vector. The three vectors represent: brothers, fathers and grandfathers for the existing words vectors. Finally, there are four vectors: main user profile vector (U), user profile brother vector (B), user profile father vector (F) and user profile grandfather vector (GF). Then, the enriched user profile vector (UO) will be:

$$UO = U + \propto B + \beta F + \gamma GF$$

Where $\propto, \beta$ and γ are coefficients to give different importance weights for each type of vectors. This proposed method depends on the employed ontology and can improve the recommendation performance from 2 to 10%.

Hawalah and Fasli (2015) model the user profile as vectors of concepts using ODP ontology in their system. After each browsing session, the visited web pages are mapped onto the ontology using a classifier. Their system consists of three agents: adding, forgetting and deleting agents. Each agent is responsible for computing the user interests' weights and attributes for the visited concepts. The user profile contains concepts and their associated attributes. These attributes include: (i) The Status which can be positive status such as browsed-concept or confirmed-concept, or negative status such as forgotten-concept or deleted-concept. (ii) The relevance_size which refers to the degree a concept is relevant to user interests. The relevance size can be measured based on user feedback about each concept. If the user feedback is positive, then the relevance size increases, but if it is negative, then it decreases. (iii) The third attribute is the frecency which represents the interest weight that indicates how much a user is interested in a concept. (iv) Finally, the frequency attribute represents the number of web pages from the user log file have been mapped to a particular concept. Their evaluation results demonstrated that the proposed method can effectively capture user interests, adapt to the changes occurring in user behaviours and can enhance the performance of a recommender system.

Tang and Zeng (2012) use an ontology that is defined by the Science Paper Online website (Science Paper, 2012). Figure 2.3 shows the concepts of the subject "computer science" in this ontology. A user profile consists of two parts: direct interests profile part and indirect interests profile part. The key task of creating the direct interest profile part of a user lies in computing weights of concepts for the research papers that read by the user. Considering a paper's possible relevance to different concepts, the possible relevance of a keyword in the paper to different concepts is taken into account. Therefore, they assign a measurement to each concept called relevance factor. For indirect interest profile part, they innovate automatic clustering algorithm to discover semantic relations between concepts that are visited by the user.

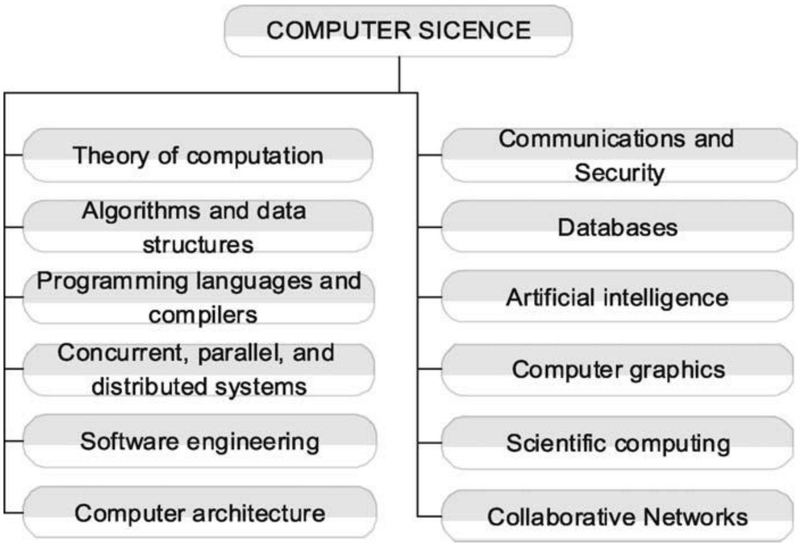

Figure 2.3. The classification of "computer science" in the Science Paper ontology (Tang and Zeng, 2012, p.89).

Kodakateri et al. (2009) designed a recommender system that recommends potential research papers of interest to users from the CiteSeer database. The 1998 version of the ACM Computing Classification System (98 ACM CCS) (ACM, 1998) is used as a reference ontology. They developed a dynamic user profile that is updated each time the user visits a new research paper. For each visited paper, the top three concepts and their corresponding weights are retrieved from the classifier. The concepts and their weight vectors are initially sorted according to the concepts' weights. If there is more than one instance of the same concept with different weights, then these weights are added together to compute the final weight associated with that particular concept in the user profile. These concepts are sorted in decreasing order. Hence, the concepts weights represent the amount of interest the user might have in a particular concept. Other examples of recommender systems that used vectors of concepts are: A Multi-Agent Personalized Ontology Profile (Gao et al., 2013) and Hypergraph-Based System (Tarakci and Cicekli, 2014).

Overall, the vectors of concepts method may be sufficient with a simple ontology that consists of two levels of classification, primary subjects and secondary subjects as shown in Figure 2.3. However, with a complex ontology such as ACM CCS ontology that maintains multiple levels hierarchy, there is a need to employ a more sophisticated technique to build a user's profile.

2.2.2.2.2 Tree of concepts

An interesting technique is developed by Chandrasekaran et al. (2008) to handle a complex ontology that maintains multiple levels of hierarchy. In this technique, a user's profile is represented as a tree of concepts. Their recommender system is for the research paper domain using the 98 ACM CCS ontology. A user's profile is created based on user's previously published research papers. In this technique, each paper is represented by a tree of concepts. A paper is entered into a classifier to determine a list of top concepts and their weights. For example, a paper D has the following concept vectors (conceptID, weight):

D= {(U, 60), (V, 40), (A, 20), (K, 10)}

Then, the concept vectors are entered into the Tree Builder Module (Lakkaraju et al., 2008) to create the tree of concepts for the paper D based on a reference tree as in Figure 2.4. The user profile is constructed as a tree of concepts that combine the concepts and their weights from all the published papers for the user.

The vectors of concepts techniques (see section 2.2.2.2.1) assume that the elements of the vectors being compared are independent, which is not accurate (Chandrasekaran et al., 2008). In order to exploit the relationships between concepts in multi-levels of hierarchical ontology such as the ACM CCS, it is more efficient to use the tree of concepts technique, because the tree of concepts technique can exploit inter-relationships between the concepts in an ontology (Chandrasekaran et al., 2008).

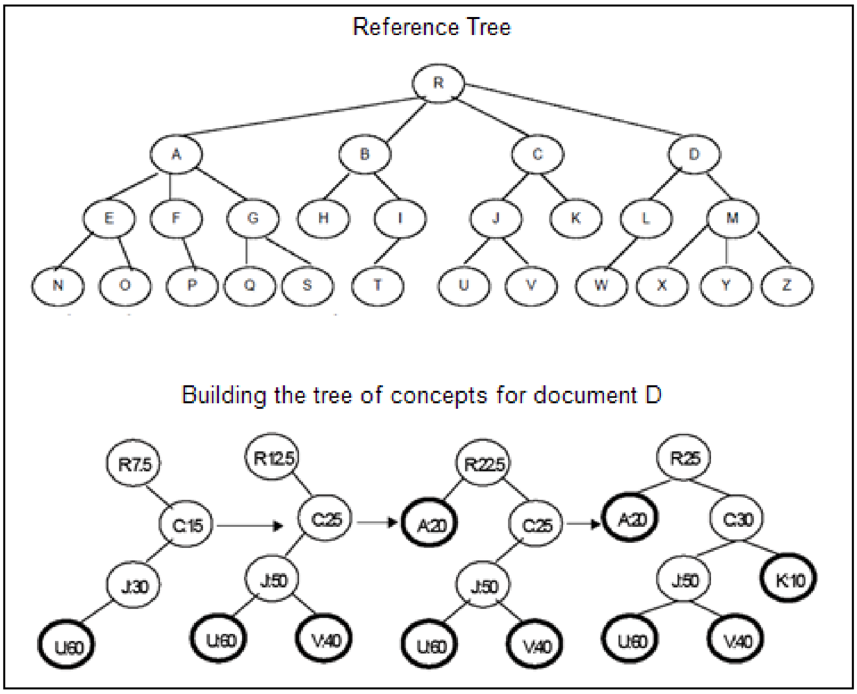

Figure 2.4. Tree of Concepts Technique (Lakkaraju et al. 2008, p2).

This is useful while computing the similarity between a user's profile and the papers' profiles (Chandrasekaran et al., 2008). However, this tree of concepts technique is static over time, whereas user preferences and needs are not static, and they usually change over time. Moreover, this user profiling technique does not normalize the concept weights in the user's tree. Without normalization, the weights in the user's tree of concepts are too big to compare accurately with the weights in a tree of concepts for a paper during the recommendation phase.

Table 2.2 summarize the user's profile representations and techniques in our literature review. However, most of these studies are not compatible with the new requirements of advanced ontologies; that become more complex and with deeper levels. Ontologies further split their original concepts into sub-concepts (Ye et al., 2010). For example, network protocol C.2.2, as a concept in the ACM CCS ontology, has been extensively studied and derived more than 70 protocols in different layers of OSI model (Ye et al., 2010). While ontologies dynamically grow, finding relevant items related to user interests is a challenging task for recommender systems. Furthermore, users may have short-term and long-term preferences during their short and long term goals. In the following section, we present dynamic techniques to build a user's profile with short-term and long-term preferences.

Reference	User profile representation	Use ontology	User profiling technique	Application domain
Lee et al., 2013	Weighted keywords	No	Bag of words	Research papers
Jomsri et al., 2010	Weighted keywords	No	Self-defined tag-based method	Research papers
Zeb and Fasli, 2011	Weighted keywords	No	Probabilistic method	News
Oh et al., 2014	Weighted keywords	No	Deep neural network	News
Alhabashneh et al., 2015	Weighted keywords	No	Fuzzy logic approach	Documents (Enterprise search)
Agarwal and Singhal, 2014	Vectors of concepts	Yes (OWL)	Dynamic user profiles	News
Fanaee-T and Yazdi, 2011	Vectors of concepts	Yes (Wordnet)	Enriched user profile with ontology	Online forums
Hawalah and Fasli, 2015	Vectors of concepts	Yes (ODP)	Multi-agent system	Web pages
Tang and Zeng, 2012	Vectors of concepts	Yes (Science Paper)	Clustering	Research papers
Kodakateri et al., 2009	Vectors of concepts	Yes (98 ACM CCS)	Dynamic user profiling	Research papers
Chandrasekaran et al., 2008	Tree of concepts	Yes (98 ACM CCS)	Static tree of concepts	Research papers

Table 2.2. An overview of user's profile representations and techniques.

2.2.3 Short-term and long-term techniques

One of the main challenges in the current recommender systems is that the user's preferences and needs are not static, they usually change over time and can be divided to short-term and long-term preferences. The short-term preferences are the recent interests of a user, which are active and require fast updating methods, whereas the long-term preferences are more stable compared with the short-term preferences (Gauch et al., 2007). Oh et al. (2014) propose a model that is based on Deep Neural Network technique for the news domain that considers long-term user interests. The words are extracted from a set of news articles which were seen by the user. Then, every word is classified into keywords or non-keywords using three layers perceptron of a deep neural network as shown in Figure 2.5. The deep neural network is used because it has a capability of adaptive learning to track changes in user preferences. There is input factor, Cumulated Preference (CP) weight, which is long-term interest weight, nonetheless, this method does not have a short-term model.

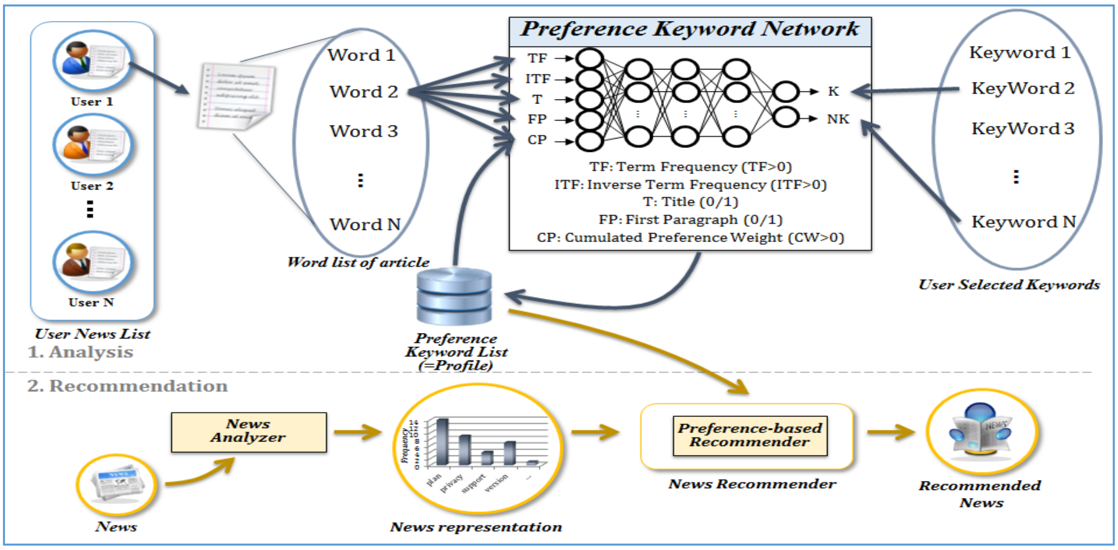

Figure 2.5. Deep Neural Network Technique (Oh et al., 2014, p1284).

Some research focuses on both short-term and long-term models. Zeb and Fasli (2011) propose a technique that constructs a probabilistic user profile that evolves according to short-term and long-term preferences. This model adapts the user profile according to the changing user interests automatically and introduces new interests to the profile and removes the non-interesting items from the profile. The long-term interests in this model are the stable interests during the users clicks on the recommended news pages, whereas the short-term interests will disappear after a while if the user gives a negative feedback (if the user did not click on the recommended news). Feedback factors are used for positive click response (αc_t) and negative click response (βc_t), where c_t is a defined constant. These two feedback factors are used in updating function for term *t* as follows. If recommendations associated with term *t* is clicked by the user, then the positive feedback factor is increased in updating function: f ($\alpha c_t +1$, βc_t). If the user did not click on the recommendations for term *t*, then the negative feedback factor is increased in the updating function: f (αc_t, $\beta c_t +1$). Stability of user's profile terms is derived from the change in click response to a recommendation in a specific period of time. That is, the stability of a term *t* with a positive click response αc_t and negative click response βc_t over a time period ΔT is computed as term stability = $\frac{\propto c_t - \beta c_t}{\Delta T}$. If the term stability ≈ 0, the user interest for term *t* tends to be stable over the time period; if term stability > 0, the user interest for term *t* tends to increase during the time period; if term stability < 0, the user interest for term *t* tends to decrease during the time period. Zeb and Fasli improved their technique in (Zeb and Fasli, 2012) to be a more time-sensitive technique that is able to speedily capture the users' interests in the news during hours by using Non-homogenous Poisson Process. Li et al. (2014) propose another recommender system in news domain that uses short-term and long-term technique. Their long-term profile for a user is constructed based on a time sensitive weighting scheme (Ding and Li, 2005). Once the long-term profile for a given user is obtained, the short-term profile is deduced about a user's recent preference. The latest read news is chosen to be the short-term goal. Li et al. (2014) stated that the reason behind choosing the latest read news lies in the fact that the latest preference can represent the user's current reading interest.

The boundaries between the short-term interests and long-term interests are defined in different ways. Agarwal and Singhal (2014) present a user profiling system that consists of independent short-term and long-term models. They suggested that analysis for short-term interests is based on the duration of past current 15 days and that of long-term interests is based on past 3 months. Term's weight is given based on a number of clicks in a session, recency of session and active session duration for short-term interests. The long-term interests are based on the click frequency. Hawalah and Fasli (2015) propose short-term and long-term models that are not based on fixed days duration. They argued that user's activities change from one day to another, for example, a user might be interested in a large range of topics in one day, but in another day he/she might have less interesting topics or even no interests at all. Moreover, not all users have the same browsing habits. Hence, depending on fixed days duration is not adequate. Building adaptive user profile for each user is complex and requires multiple tasks such as tracking the user's behaviour, adding, updating and deleting user interests. Therefore, Hawalah and Fasli (2015) proposed a multi-agent approach to create an adaptive user profile that tracks user browsing behaviour implicitly in order to extract short-term and long-term user interests. The multi-agent system addresses the complexity of user profiling by dividing large problems into sub problems, which are then managed by the individual agents as shown in Figure 2.6. They designed independent short-term and long-term models that adapt to various users' browsing behaviours. The short-term model calculates a threshold that is used to determine the concepts that should be considered as short-term interests based on user frecency behaviour on each web page. The frecency is based on reading duration. The long-term model stores all the interests that are confirmed with time and is based on concepts frequency. Unlike the short-term process which runs session by session, the long-term process should be run after a long period such as once or twice a month. Quadrana et al. (2017) address the problem of personalizing session based recommendation. Sessions from a user can occur on the same day, or over several days, weeks, or months. They assume that the short-term duration is ≤ 6 sessions and the long-term duration is more than 6 sessions. They propose a model based on hierarchical recurrent neural networks. Their results show that using short and long term models provide large improvements in recommendations comparing with one session-only recurrent neural networks recommendation. Kacem et al. (2014) propose

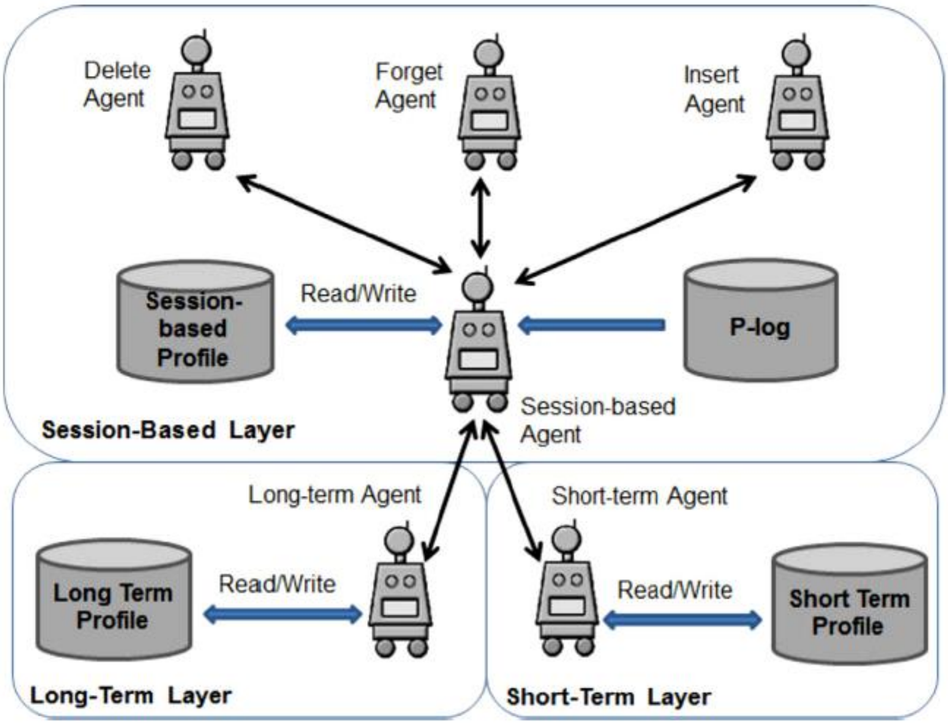

Figure 2.6. The multi-agent system (Hawalah and Fasli, 2015, p.2555).

a recommender system that uses tweets from Twitter to create a user's profile, where keywords that belong to user interests are implicitly inferred from his/her activities on Twitter. The weighting of keywords in the user profile is according to the appearing time in addition to the frequency. They adjust the importance of each keyword according to the time of its use (short-term model), unlike non-time-sensitive approach that does not consider the time but only the frequency (long-term model). The main idea is revising the notion of frequency by adjusting it with a temporal function (Kacem et al., 2014). They assume that with this method, they ensure a unified profile (combine short-term profile and long-term profile into a single one) that gives more importance to the recent interests without neglecting the continuous ones. In fact, the frequent interests may not reflect the current user need.

Short-term and long-term models also exist in modern personalization search engines. Personalization search engines can create and use a user's profile to provide him/her with interesting results when a user starts a new session of search (Leung and Lee, 2010). Gao et al. (2013) develop a personalization search system to provide personalized results to each user by using a dynamic updating policy which considers the change of the users' preferences over time and domains. The weight of each concept in the user preferences profile is dynamically recalculated with the two different kinds of strategies: window-time-based update strategy for the short-term model and time-based-forgetting update strategy for the long-term model. For window-time-based update strategy, only web pages the users have browsed during the last N days are considered. Then, the results from window-time-based update strategy are used in time-based-forgetting function update strategy for the long-term model. Moreover, the last updating time of a concept is considered. Then, the short-term model and the long-term model are combined to one complete user profile that gives different importance weights to the long-term references and short-term references. Another personalization search system is developed by Tamine-Lechani and Boughanem (2008), which re-ranks user search results by using matrix and Bayesian graphs. It has a short-term model that represents user's goals achieved within a limited number of search sessions, using a set of weighted keywords and user feedback at each retrieval session. Moreover, it has a long-term model that represents user's history that learned by managing the short-term interests and comparing the current short-term interests with previous one using Kendall rank-order correlation operator (Tamine-Lechani and Boughanem, 2008). Furthermore, Li et al. (2007) designed dynamic adaptation strategies for a personalized search system. They designed independent models for short-term and long-term user preferences, however their strategies ensure that the inherent correlations between them are not ignored, and that the changes of the short-term model have an even influence on the long-term model. For long-term model, when a user clicks on a web page (the web page classified into a topic), simply the number of times the topic has been visited increased by +1. This value is called the "TopicCount", and represents the degree of preferences. The deleting or reducing operation is affected by the changes in the short-term model. For short-term model, the Least Frequent Used Page Replacement algorithm (Li et al., 2007) is used to add and replace web pages in the user profile. Bennett et al. (2012) assessed how short-term model and long-term model interact, and how each can be used in isolation or in combination to optimally contribute to the search personalization. They found that the long-term model provides substantial benefits at the start of a search session. The short-term model provides benefits in an extended search session. The combination of the short-term model and long-term model outperforms using either alone.

Table 2.3 summarize the short-term and long-term techniques that are discussed in our literature review. Although all of these studies attempted to provide a dynamic model of a user profile by distinguishing between long-term and short-term user interests, these studies have been developed for domains such as web pages and news articles, where a user reading behaviour is different from the research paper domain. These models depend on continuous time-based user behaviour measured in days or in hours. At the present time, there is no recommender system for the research paper domain that considers short-term and long-term interests during user profile modelling. Therefore, there is a need to design a recommender system that is able to provide short-term and long-term recommendations for the research paper domain.

Reference	Short-term and long-term techniques	Combined or independent?	Application domain
Zeb and Fasli, 2011	Probabilistic method	Combined	News
Zeb and Fasli, 2012	Non-homogenous Poisson Process	Combined	News
Hawalah and Fasli, 2015	Multi-agent system	Independent	Web pages
Quadrana et al., 2017	Hierarchical recurrent neural networks	Combined	E-commerce
Oh et al., 2014	Deep neural network	Only long-term	News
Li et al., 2014	Time sensitive weighting scheme	Independent	News
Agarwal and Singhal, 2014	Mathematical method	Independent	News
Kacem et al., 2014	Time sensitive approach	Combined	Twitter
Gao et al., 2013	Multi-agent approach	Combined	Web pages
Tamine-Lechani and Boughanem, 2008	Matrix and Bayesian graphs	Combined	Web pages
Li et al., 2007	Topic count and Least Frequent Used Page Replacement algorithm	Independent	Web pages
Bennett et al., 2012	Time-weighting functions	Combined	Web pages

Table 2.3. An overview of the short-term and long-term techniques.

2.3 Recommendation approaches

Once users' profiles are modelled, recommender systems are ready to exploit these profiles using different approaches to provide recommendations. There are three main approaches for recommender systems: content-based filtering, collaborative filtering and hybrid system (Bobadilla et al., 2013 and Beel et al., 2016). Content-based filtering approaches recommend items to a user based on the similarity of the user profile and features of the items. Collaborative filtering approaches use preferences of similar users as a basis for the recommendation. Hybrid approaches combine content-based filtering and collaborative filtering methods to be more effective. In the following, we present each approach and provide examples of studies that employ each of these approaches.

2.3.1 Content-based approaches

Content-based filtering approaches recommend items to a user according to the similarity of the user profile and features of the items (Beel et al., 2016). Content-based filtering approach is based on the idea that users are interested in the items that are similar to the ones they already were seen. Each item is represented by a content profile that contains the items' features. The user profile typically is the union of all browsed items' profiles. There are items' profiles of the recommendation candidates and those candidate profiles that share most features with the user's profile are recommended (Beel et al. 2016). In this section, we present some of the current recommender systems that use content-based filtering.

Jomsri et al. (2010) designed a recommender system that compares a user's profile with papers profiles by using the cosine similarity mechanism. A user profile is built using a self-defined tag-based method and papers profiles are constructed using the TF-IDF scheme. The recommended papers are those that have a cosine similarity value greater than a threshold value as shown in Figure 2.7. In (Hawalah and Fasli, 2015; Fanaee-T and Yazdi, 2011; Agarwal and Singhal, 2014 and Oh et al., 2014) the cosine similarity technique is used as the recommendation method. In Fanaee-T and

Yazdi (2011), the cosine similarity technique is used to compare the users' profile with profiles of items from the domain and suggest those items of interest to the user according to his/her preferences. Another example of using the cosine similarity technique is that of a news recommender system developed by Oh et al. (2014). After calculating the cosine similarity values between a user's profile and all news profiles in their system, the news are ordered according to their similarity degree and the top ten news are recommended to the user.

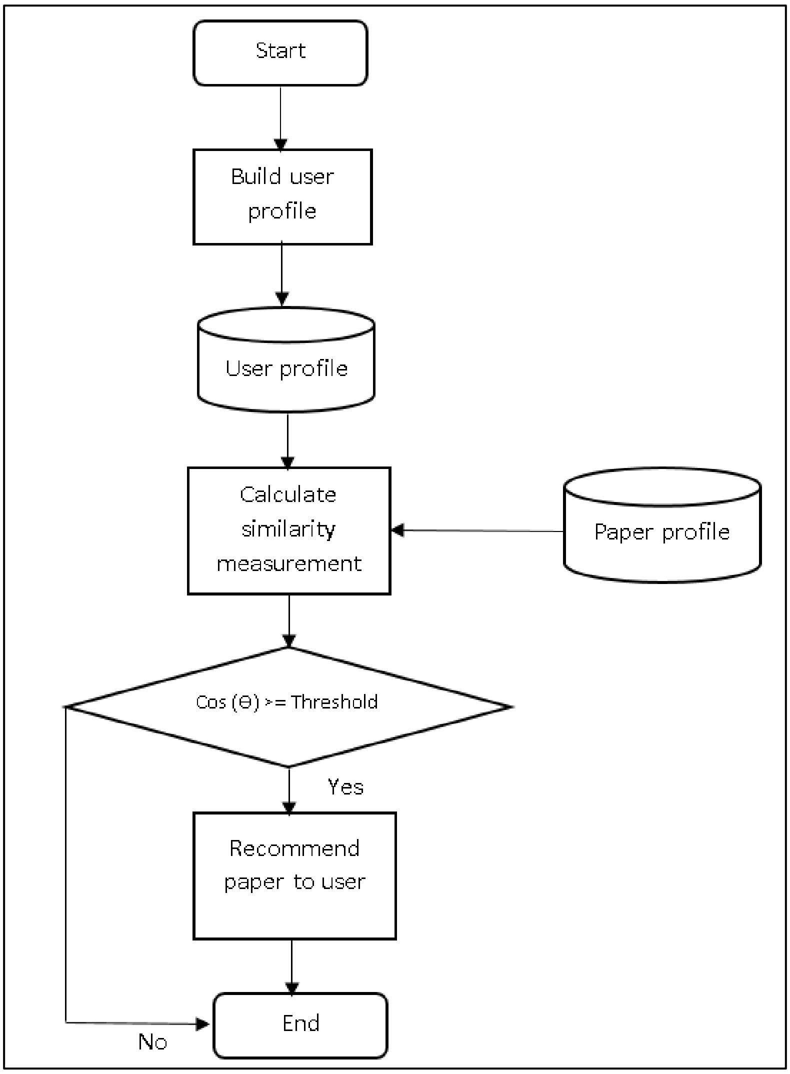

Figure 2.7. The cosine similarity with the threshold recommendation mechanism.

Although the cosine similarity technique is simple to implement, it assumes that the elements of the vectors are independent (Chandrasekaran et al., 2008). In order to exploit the relationships between concepts in multi-levels hierarchical ontology such as the ACM CCS, it is more efficient to use the proposed method by Chandrasekaran et al. (2008). They propose a recommender system that uses the tree edit distance technique to recommend interesting research papers to the user. A tree edit distance algorithm (Lakkaraju et al., 2008) is used for determining the similarity between a user profile and the paper profiles. The user profile and the paper profiles are constructed as a tree of concepts. The tree edit distance cost is the cost of transforming one tree into another with the minimum number of operations. There are three types of operation:

1. Insertion: the cost of inserting a new concept into the tree.
2. Deletion: the cost of deleting an existing concept from the tree.
3. Substitution: the cost of changing a concept's weigh to another weight.

The tree of concepts and edit distance tree techniques can exploit inter-relationships between the concepts in an ontology (Chandrasekaran et al., 2008). This is useful while computing the similarity between a user's profile and the papers' profiles. However, their system is based on static users' profiles and fixed over time, whereas users' preferences and needs are not static but they usually change over time.

Another approach to content-based filtering is proposed by Lee et al. (2013). Their approach applies clustering and neighbour-based recommendation algorithm to recommend related research papers for each user. The user may have more than one published paper with different interests, hence clustering algorithm (Lee et al., 2013) is applied and every recommended paper is assigned to only one cluster of the most similar paper written by the user and then the paper is recommended to the user. However, their approach does not employ an ontology that may improve their system. Kodakateri et al. (2009) present a recommender system that involves an ontology. For each concept *cp* in the user profile, their recommender system retrieves the research papers in the CiteSeer dataset that have the concept *cp* as one of their top concepts, as determined by their classifier. These research papers are added to the list of possible recommendations. The weight of that paper is calculated as a weight associated with the concept *cp* in the user profile multiplied by the weight associated with the concept *cp* for the paper. Then, the list is ordered in decreasing order and the top five research papers are recommended to the user.

Table 2.4 summarizes content-based approaches in our literature review. Overall, most of the proposed works in content-based recommendations suffer from different limitations and challenges when it comes to modelling dynamic recommendations that can adapt to dynamic user profiles. The first challenge is associated with recommending short and long term interests which is a complex task. Without dynamic recommendations, the recommended items would not reflect an accurate representation of user's evolving interests. Another limitation is related to items' profiles, where most of the systems focus on improving user's profile without clear improvement in items' profiles. For example, Hawalah and Fasli (2015) enhance the user profile with many features to support short-term and long-terms interests as discussed in section 2.2.3, but there is no efficient enhancement on items profiles. Moreover, the current works are insufficient to gradually evolve multiple concepts of user preferences during his/her short and long term goals and recommend them to the user.

Reference	Recommendation technique	Application domain
Lee et al., 2013	Clustering and neighbour-based recommendation algorithm	Research papers
Jomsri et al., 2010	Cosine similarity	Research papers
Fanaee-T and Yazdi, 2011	Cosine similarity	Online forums
Hawalah and Fasli, 2015	Cosine similarity	Web pages
Kodakateri et al., 2009	Multiplication method	Research papers
Chandrasekaran et al., 2008	Tree edit distance algorithm	Research papers
Oh et al., 2014	Cosine similarity	News
Agarwal and Singhal, 2014	Cosine similarity	News

Table 2.4. An overview of content-based approaches.

2.3.2 Collaborative filtering approaches

Content-based approaches can capture users' current interests, then recommend a set of items that may be related to their current interests. However, content-based approaches are not able to predict users' future interests. Collaborative filtering approaches have the ability to explore potential future interests. Collaborative filtering approaches use preferences of similar users as a basis for the recommendation. The meaning of collaborative filtering was introduced by Resnick et al. (1994). The basic idea of Resnick was that users like what like-minded users like (Beel et al., 2016). That means, when two users like the same items, they are considered like-minded. Once, two like-minded users are determined, items liked by one user are assumed to be liked by the other user and are recommended to the other user, and vice versa. The users' profiles and their ratings are required in collaborative filtering in order to find users who share similar interests. Compared to the content-based approaches, collaborative filtering approaches are different in three ways. First, collaborative filtering approach does not depend on item's profile, and depends only on the connections, which are users' ratings (Schafer et al., 2007). Second, real quality assessments of items are available due to the users' explicit ratings (Dong et al., 2009). Finally, collaborative filtering approach provides serendipitous recommendations because recommendations are based on user similarity but not on item similarity (McNee et al., 2006).

The collaborative filtering approaches are widely used in movie and e-commerce domains. There are two major categories of collaborative filtering approaches: the *memory-based* and *model-based* approaches (Shi et al., 2014 and Isinkaye et al., 2015). In the *memory-based* techniques, a user-item rating matrix is given, then a technique predicts a user's rating on a target item by combining the ratings that similar users have previously given to that item (Shi et al., 2014 and Isinkaye et al., 2015). Memory-based techniques run on entire database of ratings collected by the seller or service provider such as Amazon (Zhang et al., 2014). Usually the Pearson correlation coefficient (Benesty et al., 2006) or the cosine similarity techniques are used to identify the similar users (Shi et al., 2014 and Zhang et al., 2014). These two techniques are applied to rating vectors, each containing ratings of items in the collection that have been assigned by a target user. The *k* nearest neighbours, specifically the *k* users with the highest similarities to the target user, are selected and their ratings on the target item are combined to produce a predicted rating for the target user on that item. The Pearson correlation coefficient and the cosine similarity techniques are very simple, but they have a limitation that they consider only the co-rated items (Zhang et al., 2014). This limitation may lead to a problem that two users could have a high similarity only because they have few co-rated items and coincidently ranked these items similarity. Consequently, Ma et al. (2007) propose adding a correlation significance weighting factor that could undervalue similarity weights that were based on a small number of co-rated items. In addition to the above techniques, Gori et al. (2007) and Fouss et al. (2007) propose similarity measures by using a random-walk graph technique.

Model-based approaches are different than memory-based approaches. It first uses the ratings in the user-item matrix to train prediction models and then these trained models are used to generate recommendations for the users (Ekstrand et al., 2011). In general, the model-based approaches usually have higher accuracy than the memory-based approaches (Zhang et al., 2014). There are many techniques that are used in the model-based approaches such as feedforward neural network technique in (Vassiliou et al., 2006) and matrix factorization technique in (Gordon et al., 2008). The most representative techniques among the model-based approaches are the matrix factorization techniques (Zhang et al., 2014). Over the past studies, a lot of matrix factorization techniques have been proposed. For example, relational learning via collective matrix factorization (Singh and Gordon, 2008), probabilistic matrix factorization technique (Mnih and Salakhutdinov, 2008), extended tensor factorization technique (Moghaddam et al., 2012) and explainability constrained matrix factorization technique (Abdollahi and Nasraoui, 2017).

Nevertheless, the existing collaborative approaches are not appropriate for the research paper domain because they depend on a large number of user ratings, where there is a lack of rating in the research paper domain (Yang et al., 2009 and Beel et al., 2016). Nadee et al. (2013) tried to solve the lack of users' rating problem in book recommendation domain. They presented a recommendation approach that considers both the similarity between users and items, and items' popularity to overcome the overspecialization problem. However, their recommendation results are not sufficiently effective for the research paper domain. Therefore, there is a need to develop a new collaborative filtering model that does not depend on user ratings.

2.3.3 Hybrid approaches

Hybrid approaches combine previously introduced approaches: content-based filtering and collaborative filtering, to exploit merits of each one of these approaches (Bobadilla et al., 2013 and Beel et al., 2016). Hybrid approaches are used to improve the performance of content-based filtering and collaborative filtering (Bobadilla et al., 2013). Collaborative filtering improves content-based approaches by acquiring feedback from users (Bobadilla et al., 2013). Content-based approach improves the quality of the predictions in collaborative filtering because it is calculated with more information about items (Bobadilla et al., 2013).

Collaborative filtering and content-based approaches can be combined in four different ways as shown in Figure 2.8 (Bobadilla et al., 2013). The first way, collaborative filtering (CF) and content-based filtering (CBF) recommendations are calculated separately and subsequently combine them as shown in Figure 2.8 (A). For example, Agarwal and Singhal (2014) use collaborative filtering as well as content-based filtering for news recommendations. News in their system has two main features news source (such as BBC) and news category. Content-based filtering is used for identification of the category of user's interest. Collaborative filtering is used for recommending news source. The second way, CBF characteristics are incorporated into the CF approach as shown in Figure 2.8 (B). For example, Li et al. (2014) employ content-based filtering by using user-item affinity graph based on both long-term and short-term user's profile, and then absorbing random walk model (Zhu et al., 2007) as collaborative filtering to recommend news to a user. The third way, a unified model is constructed with both CBF and CF characteristics as shown in Figure 2.8 (C). For example, De Campos et al. (2010) employ Bayesian networks to combine CBF and CF characteristics and generate more efficient recommendations in the movies domain. The fourth way, CF characteristics are incorporated into a CBF approach as shown in Figure 2.8 (D). For example, Soboroff and Nicholas (2005) use a latent semantic index to create the users' profiles used in content-based recommendations beginning with the collaborative filtering rating matrix.

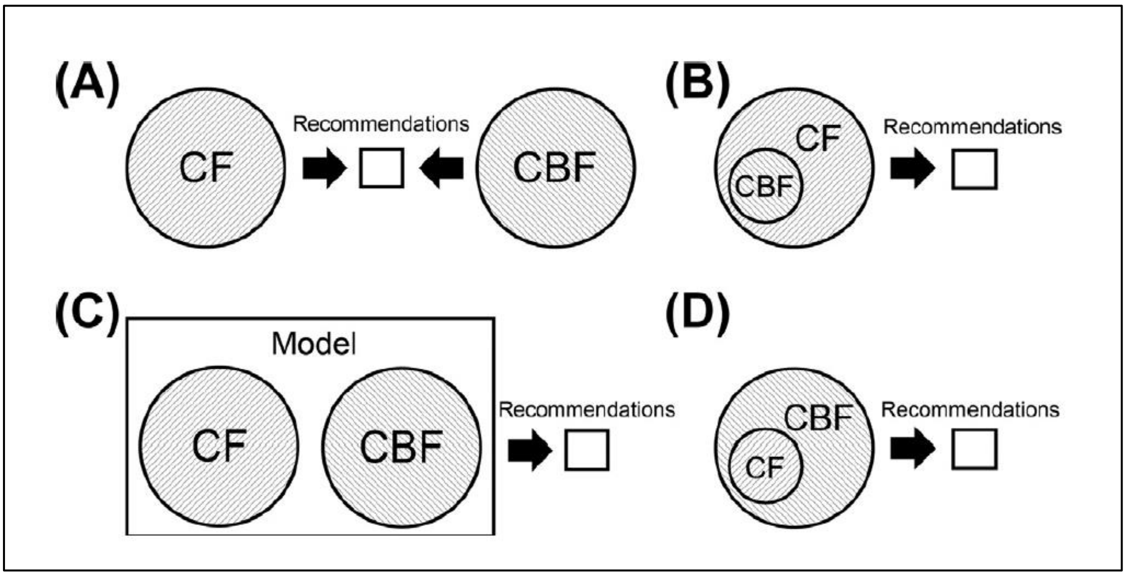

Figure 2.8. Different alternatives ways in hybrid approaches (Bobadilla et al., 2013, p122).

2.4 Gap in knowledge

Over the last few years research has been conducted to address user profiling and recommendation issues. However, there are a number of limitations in the proposed techniques. Firstly, they are not compatible with the new requirements of advanced ontologies; that become bigger, more complex and with deeper levels. Ontologies are further split for their original concepts into sub-concepts (Ye et al., 2010). While ontologies are dynamically growing in size, finding relevant research papers related to users' interests becomes a challenging task for recommender systems. Secondly, there is inefficiency to gradually evolve multiple concepts of user preferences during his/her short and long term goals. The importance of this stems from the need to design automatically adaptable user profiling technique that should keep track of multiple information that are needed by the user. The existing techniques for the research paper domain are not appropriate for the real-life rapid evolution of the user profile, where the fast deviating in multiple short-term interests may remain undetected and multiple stable long-term interests may not be changed properly according to the most recent user's preferences. Therefore, there is a need for user profiling and recommendation techniques that automatically adapt to the diverse and frequently changing of user interests and preferences. Thirdly, there is no appropriate collaborative filtering approach for the research paper domain to predict users' future interests. The existing collaborative filtering approaches have been developed for other domains such as movies and e-commerce products. These approaches depend on large numbers of user ratings. However, there is a lack of ratings in the research paper domain (Yang et al., 2009). Therefore, a collaborative filtering model is needed to discover the users' future interests. Finally, integrating a content-based model with a collaborative model to generate a hybrid recommender system also poses another challenge in the recommender systems. In general, most of the introduced models in the documents domains focus on a specific and isolated problem. They focus on addressing just one problem in recommender systems and ignore other problems. For instance, a system might focus on the current long-term user interests, but not the short-term interest, or it might focus on both long-term and short-term interests, but not the user's future interests. Therefore, such systems have just very limited recommendation capabilities. It is important to integrate all the types of user interests into a dynamic hybrid system to recommend the right preferences at the right time and rank the recommendation list according to the user needs.

Chapter 3. Evaluation Methodologies and Metrics

The evaluation methodologies in the field of recommender systems can be divided into three types of strategies: offline evaluations, user studies and online evaluations (Shani and Gunawardana, 2011 and Beet et al., 2015). These types of the methodology are illustrated in section 3.1. Then the evaluation metrics that are used to evaluate a recommender system's performance are explained in section 3.2. Finally, our evaluation methodology is presented in section 3.3.

3.1 Evaluation methodologies

3.1.1 Offline evaluations

In offline evaluations, there is no user involvement, whereas in user studies their users are employed to test the performance of the system (Shani and Gunawardana, 2011). One of the advantages of offline evaluations is that performance of the system can be evaluated at a low cost in terms of time and effort (Shani and Gunawardana, 2011). There are two situations in offline evaluations: offline evaluations using existing datasets with real users' records and offline evaluations using simulated users.

An offline evaluation using an existing dataset with real users' records is performed by using a pre-collected dataset of users choosing or rating items (Shani and Gunawardana, 2011). For example, Agarwal et al. (2005) and Vassiliou et al. (2006) used the MovieLens dataset (MovieLens, 2005), which is provided by the GroupLens Research Project (GroupLens, 2005). In order to evaluate system performance offline, it is necessary to mimic the online process where the system makes recommendations or predictions, then the user uses the recommendations or corrects the predictions. This is usually done by recording historical user data, and then hiding some of these interactions in order to mimic the knowledge of how a user will rate an item, or which recommendations a user will act upon (Shani and Gunawardana, 2011).

However, the access to a rich user dataset with real users' records that can be used for documents domains may not be available. Therefore, user behaviour simulation approach is required. Hawalah and Fasli (2011) built their own dataset using ODP and created a set of tasks that simulate five user behaviours (scenarios). Zeb and Fasli (2011) created their own dataset for news by using the RSS news aggregator and simulated user behaviours using a probability method. Nanas et al. (2009) used a dataset for news that is collected by Carnegie Group and Reuters (Carnegie Group and Reuters, 2009). They performed simulated experiments with virtual users. The advantage of using simulated users over experiments involving real users is that they are controlled and reproducible (Nanas et al., 2009). Using user simulation reduces the need for expensive user studies and online testing (Shani and Gunawardana, 2011). However, designing user simulations should be done with care. First, user simulation is a difficult task because there are different behaviours which the users exhibit. Second, if the user simulation is inaccurate, then the performance system may be optimized in simulation which has no correspondence with its actual performance in practice (Shani and Gunawardana, 2011). Though, it is reasonable to evaluate a newly designed algorithm for a recommender system using the user simulation approach to predict the performance of the algorithm at a low cost. Through these performance evaluations the algorithm can be improved before conducting expensive user studies or online evaluations.

3.1.2 User studies

A user study is conducted by using real users, and asking them to interact with a recommender system, then report and evaluate their experience (Shani and Gunawardana, 2011). While the users interact with the system, their behaviour can be observed and recorded to collect any measurements, such as the accuracy of the recommendation results. Qualitative questions can be asked before, during, and after the user interaction with the system is completed. An example of user study evaluation is to test the influence of a recommendation algorithm on the browsing behaviour of users of research papers (Shani and Gunawardana, 2011). In this example, the users are asked to read a set of research papers that are interesting to them. In some cases, this includes research papers which have been recommended and some which have not been recommended. The number of papers which were recommended and read by the user are analysed along with other data such as the number of times a recommended paper was clicked.

User studies can perhaps measure the widest set of performance evaluations (Shani and Gunawardana, 2011). Unlike offline evaluations, user studies allow us to test the behaviour of users when interacting with the recommender system, and the influence of the recommendations on user behaviour. During the offline case, assumptions such as "given a relevant recommendation the user is likely to use it" is made, which are tested in the user study (Shani and Gunawardana, 2011). User studies allow us to collect qualitative data that is often crucial for interpreting the quantitative results. Moreover, user studies allow us to use the questionnaires; users can be asked questions about their experience prior, during, and after they perform their tasks. These questions can provide information about properties that are difficult to measure, such as whether the user finds the system ease of use. However, user studies are very expensive to conduct; finding a large set of participant users and asking them to perform a large enough set of tasks is costly in terms of either user time, if the users are volunteers, or in terms of payment if paid users are employed (Shani and Gunawardana, 2011).

The number of participates in user studies for document domains varies between 3 to 35 users. For example, Jomsri et al. (2010) invited three Ph.D. students as the experiment participants. Kodakateri et al. (2009) included seven volunteer graduate students. Chandrasekaran et al. (2008) conducted a user study involving eight users from the computer science and computer engineering departments. Tang and Zeng (2012) performed an experiment with ten master students and the experiment lasted for 30 days. Uchiyama et al. (2011) included 16 graduate students. Lee et al. (2013) conducted a user study involving 30 users divided as follows: 10 programming language researchers, 10 Human-computer interaction researchers and 10 Database researchers. Hawalah and Fasli (2015) invited 30 participants with a computer science background to participate in their experiment during a 20 day period. Alhabashneh et al. (2015) used a dataset for enterprise search called TREC Enterprise 2007 Track (Bailey et al., 2007). They extended the dataset by creating search tasks and invited 35 users to participate in their study.

3.1.3 Online evaluations

During online evaluations, a system is essentially deployed and large scale of real users interact with the system, then the performance of the proposed technique is evaluated in a real environment (Shani and Gunawardana, 2011). In many realistic recommendation applications, the designer of the system may wish to influence the behaviour of users and measure users' behaviours when they interact with the recommendation system. For example, if the users of the system follow the recommendations more often, then it can be concluded that the system performs well in a real environment. The real performance of the recommendation system depends on many factors such as the user's intent (for example, how specific their information needs are), the user's context (for example, what items they are already familiar with) and the interface through which the recommendations are presented (Shani and Gunawardana, 2011). Thus, online evaluations measure the true value of a recommender system, where the system is used by real users that perform real tasks. It is better to run an online evaluation last, after an offline evaluation provides evidence that the performance of the algorithms is reasonable, and after a user study that measures the user's attitude towards the system (Shani and Gunawardana, 2011). This gradual process reduces the risk of causing significant user dissatisfaction, which may be unacceptable in commercial applications.

3.2 Evaluation metrics

In order to evaluate recommender systems, a wide range of metrics have been used. These measures can be classified into three categories based on the feature that is being evaluated: measuring the accuracy of rating predictions, measuring the accuracy of usage predictions, and measuring the accuracy of rankings of items (Shani and Gunawardana, 2011).

3.2.1 Accuracy of rating predictions

The main aim of recommender systems that use this type of evaluation is to compute the rating for items and then recommend the highest rated items to the users. They usually evaluate the accuracy of the predicted ratings and compare them to users' actual ratings (Shani and Gunawardana, 2011). This type of evaluation is useful in recommender systems that predict ratings and recommend items to users such as music or movies recommender systems. There are two popular metrics in this type: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

Mean Absolute Error (MAE) (Jannach et al., 2010) measures the average error between predicted ratings and users' actual ratings. For a user *u* that is recommended an item *i*, the MAE is computed based on the predicted rating r_i and a user true rating t_i for the same item *i* as follows:

$$MAE = \frac{\sum_{i=1}^{N} |t_i - r_i|}{N} \qquad (3.1)$$

where *N* is the total number of items. Therefore, the system with a lower error would have better predicted ratings and hence better performance.

Root Mean Square Error (RMSE) (Jannach et al., 2010) is another metric to compute the error. The main difference in this metric is that before the rating errors are summed up, they are first squared. The main objective of this metric is to highlight large errors and provide more weights to them than small errors. The RMSE for the ratings *r* of items *i* for a user *u* is computed as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (t_{iu} - r_{iu})^2}{N}} \qquad (3.2)$$

Some studies in the literature have adopted this type of evaluation. For example, Zeb and Fasli (2011) proposed a technique that constructs a probabilistic user profile that evolves according to short-term and long-term preferences. In order to evaluate this technique, they used the MAE metric to compare the performance of their recommender system against other systems.

This type of evaluation metric has some advantages and disadvantages. One important advantage is that the computation process is simple, as once a system predicts a set of ratings, the accuracy of them can be computed by comparing them to the user's true ratings. However, one disadvantage is that this evaluation is only able to measure the accuracy of systems that predict ratings, but not the systems that can recommend a list of ranked items that have no ratings. For some recommender systems, the main goal is to provide a user with a relevant list of items that might be interesting to him/her, so these metrics cannot be used as there are no ratings that are compared.

3.2.2 Measuring usage predictions

In many domains in recommender systems, computing item ratings is not important, but what is important is to predict items that users are likely to be interested in (Shani and Gunawardana, 2011). For example, in research papers and news recommender systems, it is usually important to recommend the top research papers and news that users might be interested in. The process of evaluating these kinds of systems is to compare which system provides more relevant recommendations to users, as the best system is the one that provides the more relevant items in the top N results. According to Shani and Gunawardana (2011), the classical process for this evaluation is to have a dataset that holds all the items that users have selected. A part of this dataset is usually used by the recommender system to recommend new items to users, whereas the other part is used to evaluate whether the recommended items are relevant to the users or not. Popular metrics in this type of evaluation are Precision and Recall. Precision aims to find the number of the recommended items which are relevant to users, while recall assesses the quality of the recommender system in terms of how many items are relevant to users have been successfully retrieved (Manning et al., 2008). Precision and recall are usually computed based on dividing all the items into four parts as in Table 3.1.

	Recommended	Not recommended
Relevant	TR	FN
Irrelevant	FR	TN

Table 3.1. Classification of the possible result of a recommendation of an item to a user.

The precision metric can be defined as the ratio of relevant items that are selected by users (*TR)* to the total number of items that are recommended by the system (*TR* + *FR),* and it is computed as follows:

$$Precision = \frac{Number\ of\ TR}{Number\ of\ (TR+FR)} \qquad (3.3)$$

On the other hand, recall is presented as the ratio of relevant items found to be truly relevant to users (*TR)* to the total number of available relevant items (*TR* + *FN)* and it is computed as follows:

$$Recall = \frac{Number\ of\ TR}{Number\ of\ (TR+FN)} \qquad (3.4)$$

Precision and recall provide two different metrics, hence a single measure F-measure that trades off precision versus recall can be used (Manning et al., 2008). F-measure can be computed as follows:

$$F\_measure = \frac{2 * Precision * Recall}{Precision+Recall} \qquad (3.5)$$

All the retrieved items by a recommender system are taken into consideration in the above precision and recall. In some cases, a recommender system might not retrieve and present all the items to users; instead it might just recommend the top N relevant items. Therefore, the P_N metric has been proposed to deal with just the top N results that are retrieved and recommended to users. In P_N metric, the precision is computed at different cut-off results such as the top 10, 20, or any predefined number of results. Formally, the P_N can be computed as follows:

$$P_N = \frac{Number\ of\ TR\ in\ top\ N}{N} \qquad (3.6)$$

Where *N* is the cut-off number of retrieved items and *number of TR in top N* is the total number of relevant items in the top *N* results. More versions of precision are the average precision and mean average precision metrics. In these metrics, the average of different top *N* precision items is computed as a single value. Therefore, these measures appropriate in case of comparing different recommender systems or algorithms. Formally, average precision for different cut-off results for a user is calculated as follows (Jannach et al., 2010):

$$AVG\ P = \frac{P_{N1} + P_{N2} + \cdots \ldots + P_{Nm}}{m} \qquad (3.7)$$

Where *m* is the number of different cut-off results. Mean average precision (MAP) calculates the mean average precision for a set of users. The mean average precision for *U* users is the average of the average precision of each user (Manning et al., 2008):

$$MAP = \frac{\sum_{i=1}^{U} AVG\ P}{U} \qquad (3.8)$$

Many studies in the literature have used this type of evaluation to evaluate their systems. For instance, Kodakateri et al. (2009) proposed a recommender system that recommends potential research papers of interest to users from the CiteSeer dataset. They used the precision metric to evaluate the research paper recommender system. The top five recommended research papers are collected and presented to the users. Users were asked to judge these research papers as relevant or irrelevant. Another study that used this type of evaluation is proposed by Hawalah and Fasli (2015), who proposed a multi-agent approach to create an adaptive user profile that tracks user browsing behaviour to extract short-term and long-term user interests. The evaluation process in this study relied on the precision, P_N and average precision metrics, which were used to compare different systems. The MAP is used by Nadee et al. (2013) to evaluate a collaborative recommender system for books.

Precision is an appropriate for measuring performance of systems that only aim at providing highly relevant items to users (Agarwal et al., 2005 and Hawalah and Fasli, 2015), whereas recall and F-measure are not suited for these types of systems for the following reasons. The main aim of research paper recommender systems is to present small amount of relevant information from a relatively large source of information. Therefore, it is more important to return a small number of recommendations that contain relevant items rather than giving the user large number of recommendations that may contain more relevant recommendations but also requires the user to manually select through many irrelevant results. The ratio between the number of relevant results returned and the number of truly relevant results is defined as the recall. Notice it is possible to have a very high recall by making a lot of recommendations. In the research paper domain, a user will be more interested in reading research papers that are relevant to his/her interests rather than going through a large list of recommended research papers and then selecting those which are of interest. Precision is a more accurate measure for the research paper recommender system than recall (Agarwal et al., 2005 and Hawalah and Fasli, 2015). Therefore, computing the recall and F-measure usually is not important for research paper recommender systems. According to Beel et al. (2016), the precision metric is used in 72% of the studies, the recall is used in 23% of the studies and the F-measure is used in 11% of the studies.

Overall, this type of evaluation is simple to conduct, as it only needs users to rate a few number of results as being either relevant or irrelevant to their needs. However, one limitation of this type is that it is unaffected by the results' order. That is if two systems retrieved the same items but in different rank in a list (for example, one of them presented the relevant items at the top of the list, whereas the other at the end of the list), they would be assigned the same precision results.

3.2.3 Ranking measures

In the previous two types, the main aim of the evaluation is to evaluate the accuracy of rating predictions or the retrieval process. However, these evaluations might not be appropriate to evaluate a system that aims at providing a ranked list of items where the items that are more relevant to users are placed higher on a list than those that are less important. In this case, the order of items is the main concern for such systems. Some metrics have been proposed to evaluate the ranking accuracy. One of them is Normalized Distance-based Performance Measure (NDPM) (Yao, 1995), which uses a reference ranking (a correct order). By using a reference ranking we can try to determine the correct order for a set of items for each user and measure the similarity of the system's results with the correct order (Shani and Gunawardana, 2011). In this evaluation metric, it is first necessary to obtain a reference ranking to evaluate a ranking algorithm. If explicit user ratings of items are available, then we can rank the rated items in decreasing order of the ratings, with ties. For instance, Netflix (Netflix, 2014) movies ranked by a user can be ranked in decreasing order of rating, with 5-star movies tied, followed by 4-star movies tied, until 1-star movies tied. If we only have usage data, then it is more applicable to construct a reference ranking where items selected by the user are ranked above unselected items. However, this is only valid if we know that the user was aware of the unselected items, so that we can infer that the user actually preferred the selected items to the unselected items. The NDPM measure provides score 0 to systems that accurately predicts every preference relation asserted by the reference ranking. The score of 1 is assigned to worse systems that contradict every preference relation asserted by the reference ranking (Shani and Gunawardana, 2011).

If we do not have a reference ranking, we can attempt to measure the utility of the system's raking to a user using R-Score metric (Breese et al., 1998) or Normalized Cumulative Discounted Gain (NDCG) (McSherry and Najork, 2008). The R-Score metric is used for applications where the user can use only a single or a very small set of items. In this kind of applications, the users are expected to observe only a few items of the top of the recommendations list. Hence, R-Score is suitable because it has the very rapid decay of the positional discount down the list (Shani and Gunawardana, 2011). The R-Score metric assumes that the value of recommendations declines exponentially down the ranked list to yield the following score for each user u:

$$R_u = \sum_{j=1}^{N} \frac{\max(r_{u,i_j} - d)}{2^{\frac{j-1}{\propto - 1}}} \tag{3.9}$$

Where i_j is the item in the jth position, $r_{u,i}$ is user u's rating of item i, d is a task dependent neutral ("don't care") rating, and α is a half-life parameter, which controls the exponential decline of the value of positions in the ranked list.

Normalized Cumulative Discounted Gain (NDCG) is used for applications where a user is expected to read a relatively large portion of the list, such as searching for relevant documents. In such case, a much slower decay of the positional discount is needed (Shani and Gunawardana, 2011). The NDCG is a measure from information retrieval, where positions are discounted logarithmically. Assuming each user *u* has a relevance value rel_i from being recommended an item *i*, the Discounted Cumulative Gain (DCG) for a list of *M* items is defined as (McSherry and Najork, 2008):

$$DCG = \sum_{i=1}^{M} \frac{2^{rel_i - 1}}{log_2(i+1)} \qquad (3.10)$$

The NDCG is the normalized version of the DCG given by:

$$NDCG = \frac{DCG}{IDCG} \qquad (3.11)$$

Where IDCG is the ideal DCG, which is the maximum possible (ideal) DCG for a given set of items and relevancies ordered by decreasing relevance. Moreover, recommender systems can use a cut-off top-k version of NDCG. Such NDCG measure is usually referred to as $NDCG_k$ (McSherry and Najork, 2008). Some studies have adopted this type of evaluation. For example, Sugiyama and Kan (2010) employ the NDCG metric to evaluate their system. A recommendation list is presented to a user, then the user may select a number of relevant research papers from the list. Kacem et al. (2014) use the NDCG metric to evaluate the top 10 recommended tweets ($NDCG_{10}$) for their recommender system that use data collected from Twitter.

Overall, this type of evaluation measures the ranking accuracy for a ranked list of items where the items that are more relevant to users are placed higher in a list than those that are less important. However, one limitation of the existing methods of this type is that their measures are according to single user's interest not for multiple user interests. That is, if a user interested in more than one concept (e.g. data mining and programming languages), they are unable to evaluate ranked list for both concepts (more explanation of this limitation in section 7.3). Table 3.2 summarizes the used methodologies and evaluation metrics based on our literature review for the documents domains.

Reference	Evaluation methodology	Number of users	Evaluation metrics	Application domain
Lee et al., 2013	User study	30	Precision	Research papers
Jomsri et al., 2010	User study	3	Precision	Research papers
Zeb and Fasli, 2011	Offline	NA	MAE	News
Fanaee-T and Yazdi, 2011	Online	35	F-measure	Online forums
Hawalah and Fasli, 2015	User study	30	Precision, P_N and average precision	Web pages
Alhabashneh et al., (2015)	User study	35	Precision and recall	Documents (Enterprise search)
Tang and Zeng, 2012	User study	10	Precision	Research papers
Kodakateri et al., 2009	User study	7	Precision	Research papers
Chandrasekaran et al., 2008	User study	8	Their own method using user's judgment on a scale of 1-4	Research papers
Oh et al., 2014	User study	8	Precision	News
Agarwal and Singhal, 2014	Offline	NA	Precision, recall and F-measure	News
Li et al., 2007	User study	12	MAE	Web pages
Nadee et al., 2013	Offline	NA	MAP	Books
Sugiyama and Kan, 2010	User study	28	NDCG	Research papers
Kacem et al., 2014	Offline	800	NDCG	Twitter

Table 3.2. An overview of the methodologies and evaluation metrics.

3.3 Our research methodology and evaluations

The evaluation process of recommender system algorithms is known to be difficult and expensive as these systems are typically complex and have many components, properties and parameters which have to be carefully examined in order to provide the optimum performance (Li et al., 2014). According to Beel et al. (2016), 59% of studies conduct offline evaluations, 34% user studies and only 7% online evaluations. Offline evaluations are more convenient than conducting user studies or online evaluations, because results are available within minutes or hours and not within weeks or months as is the case for user studies and online evaluations. Moreover, many researchers have no access to real-world systems to evaluate their approaches (Beel et al., 2016). Therefore, the offline evaluation methodology is conducted in this thesis. The evaluation metrics that are used to evaluate the performance of the proposed models are precision P_N, average precision, and mean average precision.

As we explained in section 3.1.1, access for a rich dataset with real users' records that can be used for documents domains may not be available. After we finished our first model (i.e. the DNTC model in chapter 4), we did not have access to a rich dataset with real users' records for the research paper domain to evaluate our model. Therefore, simulation approach that simulates user behaviour is needed. We opted to use the user behaviour simulation approach to test specific scenarios for multiple concepts and variant range of papers quantity to evaluate our DNTC model. We used research papers from the CiteSeerX dataset (CiteSeerX, 2015), and for the classifier we used the dataset from the ACM Digital Library with the 2012 ACM CCS ontology (ACM, 2012) for the field of computer and information science. We chose this ontology because ACM has a deep multilevel hierarchal ontology. Our preliminary evaluation and results are explained in section 4.5.2.1. While we are developing the short-term and long-term models in chapter 5, we had access to the BibSonomy dataset (Knowledge and Data Engineering Group, 2017) with rich users' records. The BibSonomy dataset contains actual records of users' interests as posts for research papers over approximately a ten-year period. Each post contains: metadata for a research paper, date and time of the post. We consider these posts as users' reading records of research papers. We used records of users reading behaviour over the years 2015 and 2016 for users in the field of computer and information science. This includes 1,642 users and 43,140 research papers. The DNTC model is revaluated using the BibSonomy dataset as we will discuss in section 4.5.2.2. Then, we analysed the users' records for short-term and long-term interests using the BibSonomy dataset in chapter 5. After that, we developed the novel content-based short-term and long-term models that suit the research paper domain. Afterwards, a novel collaborative recommendation method to predict the users' future interests is presented in chapter 6. In every chapter (in chapters 4, 5, 6), we developed a model that focuses on a specific problem/limitation, then the model is evaluated separately to ensure that it is able to solve that specific problem/limitation. Then, in chapter 7 we integrated all the models to be a novel dynamic hybrid system. The innovative dynamic hybrid system incorporates the content-based models for short-term and long-term interests with the collaborative model to provide a user with a recommendation list that contains the most related research papers to his/her interests in the right time. The evaluations and results for the dynamic hybrid system by using the precision metric are explained in section 7.2. Then, we evaluated the dynamic hybrid system and each of the individual systems by using our new M_NDCG ranking measure in section 7.3.

Chapter 4. Modelling Recommender System Using Dynamic Normalized Tree of Concepts (DNTC)

Current recommender systems suffer from a number of limitations that might restrain the recommendation services. One critical limitation in these systems is that they are not compatible with the new requirements of advanced deep ontologies, even the systems that consider the advanced ontologies they are modelled to be static over time. A significant problem with such modelling techniques is that they assume user needs and preferences to be static over time, however, users preferences are dynamic and change over time. Our user modelling technique overcomes this problem by dynamically capturing the user preferences and representing the user profile as a dynamic normalized tree of concepts (DNTC). In this chapter, we propose a content-based recommender system for the research paper domain that consists of three main phases: research papers classification phase, dynamic user profiling phase and the recommendation phase. The first phase is responsible for preparing research papers and classifying them. The second phase is responsible for tracking user reading activities for research papers. The research papers that are read by the user are used to build a user profile represented as a DNTC. The third phase is the recommendation phase which makes use of the dynamic tree edit distance technique to recommend a set of research papers to the user based on his/her preferences. Figure 4.1 shows an overview diagram for the proposed DNTC system. The next section discusses our ontology model. After that, our system's phases will be explained in detail in sections 4.2, 4.3 and 4.4. The evaluations and results will be discussed in section 4.5. Finally, the conclusions are presented in section 4.6.

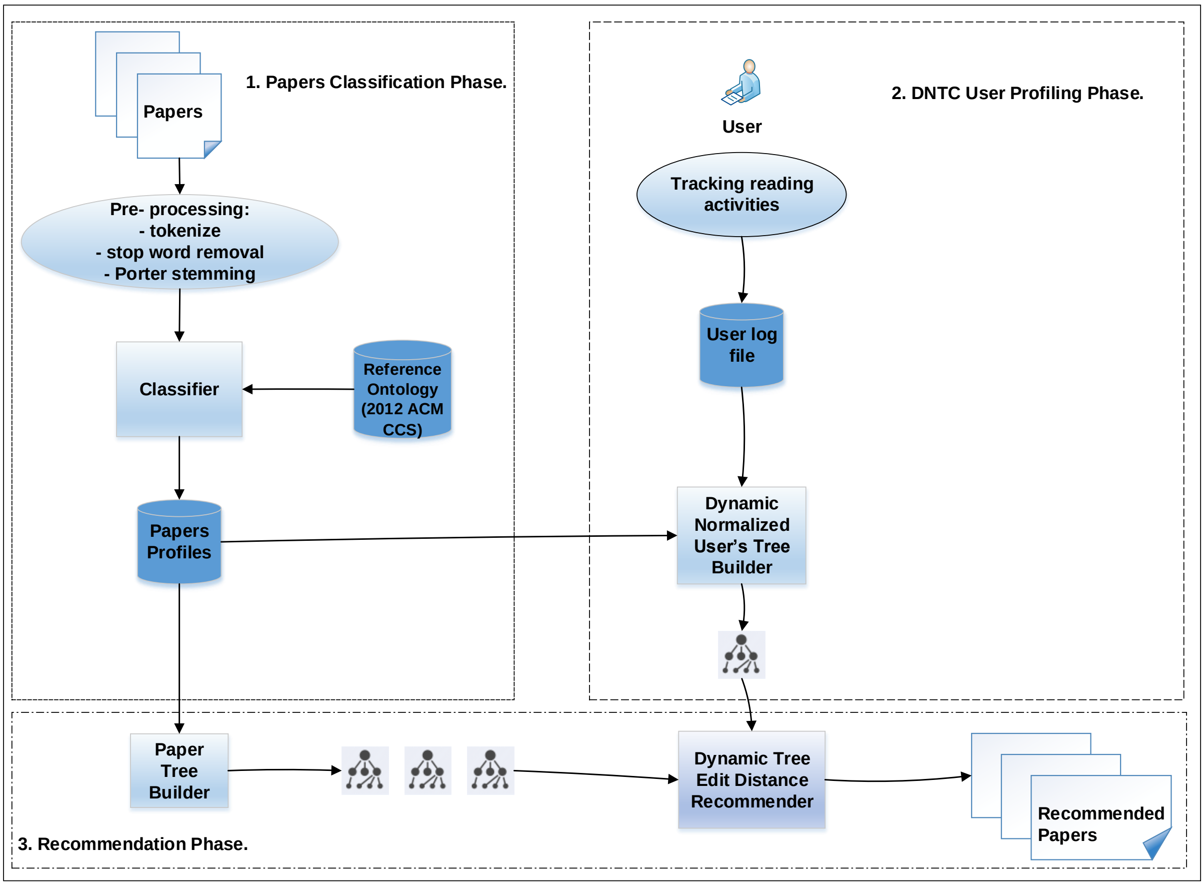

Figure 4.1. The proposed DNTC system architecture.

4.1 Ontology in our system

An ontology is a rich knowledge representation which has been shown to provide significant improvements in the performance of user profiling models (Challam et al, 2007). A reference ontology provides a clear illustration of the contents of a particular domain of an application as discussed in section 2.2.2.2. In our system, a reference ontology is used for three main purposes:

1) Mapping a paper to the correct concepts using a classification algorithm.

2) Representing a **paper** profile as a tree of concepts.

3) Representing a **user** profile as a normalized tree of concepts.

We use the ontology for the 2012 ACM Computing Classification System (2012 ACM CCS) because it maintains a deep multilevel hierarchy that can be utilized in semantic web applications. The traditional 1998 version of the ACM Computing Classification System (98 ACM CCS) is replaced with the 2012 ACM CCS. The traditional 98 ACM CCS has served as the de facto standard classification system for the computing field and some of the recommender systems used it (Lakkaraju et al., 2008, Chandrasekaran et al., 2008 and Kodakateri et al., 2009). However, new deep concepts have appeared in the computing field after 1998. Therefore, 98 ACM CCS ontology has been improved to the 2012 ACM CCS ontology to cover the new concepts. The 2012 ACM CCS is being integrated into the search capabilities of the ACM Digital Library. It relies on a semantic vocabulary as the single source of categories and concepts that reflect the state of the art of the new computing discipline and is receptive to structural change as it evolves in the future. ACM provides a tool within the visual display format to facilitate the application of 2012 ACM CCS categories to forthcoming research papers and a process to ensure that the CCS stays current and relevant. To the best of our knowledge, our recommender system is the first recommender system for research papers that uses the 2012 ACM CCS. The usage of the 2012 ACM CCS in our classification phase is explained in the next section.

4.2 Research papers classification phase

In the first phase in our system, we build a classifier using the ACM training dataset and classify a set of research papers from the CiteSeerX dataset (CiteSeerX, 2015) and the BibSonomy dataset (Knowledge and Data Engineering Group, 2017) (see section 4.5 for more information about these datasets) to the reference ontology. The research papers in the dataset are mapped to the reference ontology by classifying each paper to the correct concepts using the Term Frequency-Inverse Document Frequency (TF-IDF) technique (Dillon, 1983) and cosine similarity (Ricardo and Berthier, 2011). The classification process consists of two phases:

1) Training phase: During this phase, research papers in the ACM training set, which are pre-assigned to one or more concepts in the reference ontology, are used to learn a vector of features for each concept in the ontology.

2) Classification phase: In this phase the cosine similarity classifier uses the vectors learnt in the training phase to classify research papers in the CiteSeerX and the BibSonomy datasets. The output is a list of concepts for each input paper along with their corresponding weights which indicate the degree of association between the concept and the paper. The top N concepts for each research paper are retained and stored in the paper profile.

4.2.1 Training phase

The training set was provided by the ACM (ACM, 2012). The training set contains research papers which are pre-assigned to one or more concepts in the 2012 ACM CCS ontology manually by the authors of the research papers. The ACM dataset contains 16,307 mapped research papers for the 2012 ACM CCS ontology. The main categories in the 2012 ACM CCS are: Hardware, Computer Systems Organization, Networks, Software and Its Engineering, Information System, Theory of Computation, Mathematics of Computing, Security and Privacy, Human-Centered Computing and Computing Methodologies. The total number of the concepts under these categories is 1,329 concepts, and the number of leaf concepts is 986 concepts. The concepts in the ontology reference are associated with training research papers that represent each concept. The research papers for each concept are combined into one document (d_j) to represent a concept (c_j). Each document is tokenized and represented as a set of terms constructed from the papers' title, abstract and keywords. We applied some heuristics functions to pre-process the text, these functions are stop words removal and then Porter stemming algorithm (Sparck and Willett, 1997) which reduces each word (term) to its shortest stem. The documents are then represented as weighted feature vectors by using the TF-IDF weighting algorithm. The TF-IDF is used to determine the importance of a word in a document within a collection or corpus (the corpus in our system is the training set). The importance increases proportionally to the number of times a term appears in a document but is offset by the frequency of the term in the corpus. The TF-IDF is calculated as follows:

$$TF\text{-}IDF\ (t_{ij}) = TF(t_{ij}) * IDF_i \qquad (4.1)$$

where $TF(t_{ij})$ is Term Frequency that measures how frequently a term t_i occurs in a document d_j. Since the documents are different in length, it is possible that a term would appear more times in longer documents than shorter ones. Thus, the term frequency is normalized using the document length:

$$TF\ (t_{ij}) = \frac{Number\ of\ times\ term\ t_i\ appears\ in\ a\ document\ d_j}{Total\ number\ of\ terms\ in\ a\ document\ d_j} \qquad (4.2)$$

The IDF_i is Inverse Document Frequency which measures the importance of a term t_i across all documents in the training set:

$$IDF(t_i) = log(\frac{Total\ number\ of\ documents\ in\ the\ training\ set}{Number\ of\ documents\ with\ term\ t_i\ in\ the\ training\ set}) \qquad (4.3)$$

The TF-IDF weighted terms are calculated between 0 and 1 for each document in the training set. Therefore, all the concepts in the reference ontology are associated with training documents that have TF-IDF weighted terms, which can be used to measure a vector similarity between a concept represented by the document and a paper that we want to classify.

4.2.2 Classification phase

In this phase, research papers from the CiteSeerX and the BibSonomy datasets are classified to create databases of paper profiles for the recommender system to make recommendations from. The cosine similarity method is used to assign an input paper to appropriate concepts in the reference ontology. In our system, the cosine similarity algorithm (Ricardo and Berthier, 2011) is applied to classify an input paper to the correct concepts:

$$SW_j = CosinSim\ (d_j, P) = \frac{\sum_{i=1}^{n}(w_{ij} * w_{iP})}{\sqrt{\sum_{i=1}^{n} w_{ij}^2} * \sqrt{\sum_{i=1}^{n} w_{iP}^2}} \qquad (4.4)$$

where d_j is a document that represents a concept c_j in the reference ontology, P is an input paper, w_{ij} is the TF-IDF weight for term t_i in d_j and w_{iP} is the TF-IDF weight for term t_i in P. The cosine similarity is computed between all concepts' documents and paper P. The output from the classification phase is papers' profile for representing the research papers, composed of a decreasing ordered list of concepts' IDs along with their cosine similarity (c_j, SW_j) for each input paper P in the dataset. The cosine similarity (SW_j) is the degree of association between a paper P and a concept c_j. The resulting profile of research papers is stored in a database which is used to build the tree of concepts model for the users and the research papers.

4.3 DNTC user profiling phase

The main goal of the user profiling phase is to build the user profile as a dynamic normalized tree of concepts. Building a user profile as a tree of concepts maintains the parent-child relationships between the concepts in the ontology. These relationships can be useful while computing the similarity between a user profile and a research paper's profile. Normalizing the user's tree of concepts by the number of research papers read by the user provides a more accurate comparison between a research paper's profile and the user profile (which generally involves more than one paper).

All research papers read by a user are stored in a log file with each paper's unique identifier and the associated time sequence of each paper's reading order. Hence, the user profile is dynamically updated each time the user reads a new paper (we assume if the user reads a paper, then this is a paper of interest to the user). We added this new feature (the time sequence of the paper's reading order) to make the proposed tree profiling model dynamic and changeable because user preferences and interests change over time.

For each paper that is read by the user, the top N related concepts and their corresponding cosine similarity weights are retrieved from the paper's profile, which results from the classification phase. In order to exploit the relationships between concepts in a hierarchical concept ontology, a user tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts. Then, the user tree is updated each time a new paper is read by the user as follows. For every new paper, the top N concepts and their corresponding cosine similarity (SW) weights are used to update the existing user tree. First, the SW weights for the top N concepts are updated by adding the new SW weights to old weights values in the user tree. Then, the new weight values recursively propagate to the parent nodes until the root node is reached. We assign weights to parents according to the following equation:

$$SW_{Parent} = \propto \times SW_{Child} \qquad (4.5)$$

Where SW_{Parent} is the weight of the parent, SW_{Child} is the weight of the child and α is the weight propagation factor. α is used to maintain the parent-child relationships between the concepts in the user's tree and its value varies between 0 and 1. If α is given the value zero, then the parents will not be assigned any part of the child's weight and there will be no actual tree structure in the user profile, which means a user profile is created as a vector of concepts without any parent-child relationships in a tree structure. Otherwise, if α is given non zero value ($0 < \alpha < 1$), then a user profile will be created as a tree of concepts. α is used to determine how much of a child's weight is propagated to its parent. The value of α will be discussed in section 4.5.2.1.1.

Finally, all concept weights are divided by the total number of research papers that are read by the user in order to normalize the concept weights. Without normalizing the user's tree of concepts, the concept weights are too large in comparison to the weights in a tree of concepts for a single paper in the recommendation phase. Figure 4.2 presents our DNTC user profiling algorithm. The output of the DNTC user profiling phase is a normalized tree of concepts and their corresponding weights. This dynamic normalized tree is used in the recommendation phase in section 4.4. The following example explains our dynamic tree of concepts user profiling.

```
Build Dynamic Normalized User Tree (UserID, UserTree, PapersProfiles, CurrentTime,
Alpha, TopN)
{
  CurrentNumberOfUserPapers =0;
  Foreach Paper Pi in user's log file in CurrentTime do
   {
     CurrentNumberOfUserPapers = CurrentNumberOfUserPapers + 1;
     Get the TopN concepts and their corresponding weights from Paper Pi Profile;
     Foreach concept cj in the TopN concepts do
       {
         Find the concept cj in the UserTree;
         Update the concept cj weight: SWj += Pi_SWj;
         If the concept cj is not root do
          {
            CurrentConcpet = cj;
            CurrentConcept_SW = SWj;
            Loop until UserTree's root reached
              {
                Get currentConcpet.Parent;
                Update currentConcpet.parent weight: SWP+= CurrentConcept_SW * Alpha;
                CurrentConcpet = currentConcpet.Parent;
                 CurrentConcept_SW = SWP;
              }
          }
       }
    }

  //Divide all the concepts' weights by the current total number of user's reading
    papers.
  Foreach concept cj in UserTree do
    {
       Divide the concept cj weight: SWj = SWj / CurrentNumberOfUserPapers;
    }
}
```

Figure 4.2. The DNTC user profiling algorithm.

Example of building tree of concepts for a user:

Figure 4.3 shows a reference ontology that used for this example. We assume that a user A read three research papers P_1, P_2 and P_3. Let the top three concepts with their weights that resulted from the classification phase for the research papers P_1, P_2 and P_3 as follows:

P_1 = { (c_{122} , 0.22) , (c_{212} , 0.03) , (c_{123} , 0.02) }

P_2 = { (c_{122} , 0.37) , (c_{321} , 0.05) , (c_{322} , 0.04) }

P_3 = { (c_{333} , 0.26) , (c_{332} , 0.07) , (c_{122} , 0.03) }

We assume the weight propagation factor α is 0.25.

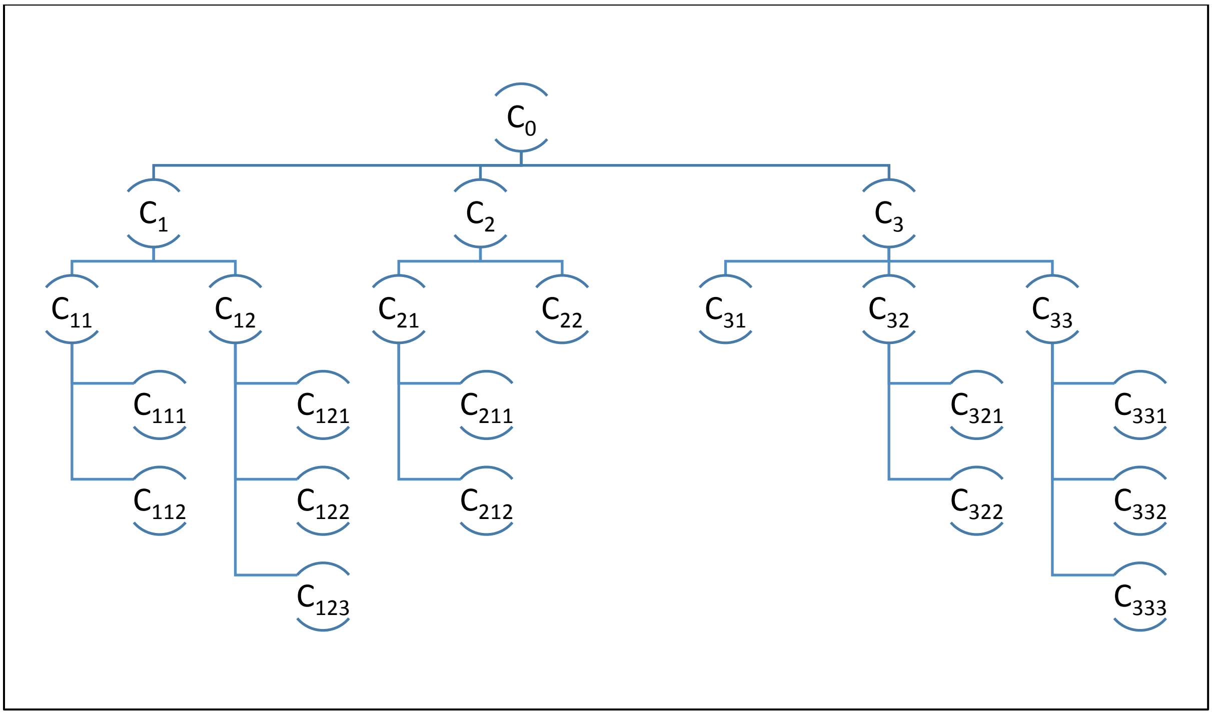

Figure 4.3. A reference ontology.

Figure 4.4 shows the user tree after reading the first paper P_1. In this figure each concept associated with its current weight (c_j, SW_j). Concepts with bold font are the effected concepts: c_{122}, c_{123}, c_{212} and their parents and grandparents until the root is reached. The weight of the concepts c_{122} and c_{123} is updated to be SW_{122} = 0.22 and SW_{123} = 0.02. Then, the weight of their direct parent c_{12} is updated according to equation 4.5 as follows:

$$SW_{12} = (\propto \times SW_{122}) + (\propto \times SW_{123}) = 0.25 \times (\, 0.22 + 0.02) \; = 0.06$$

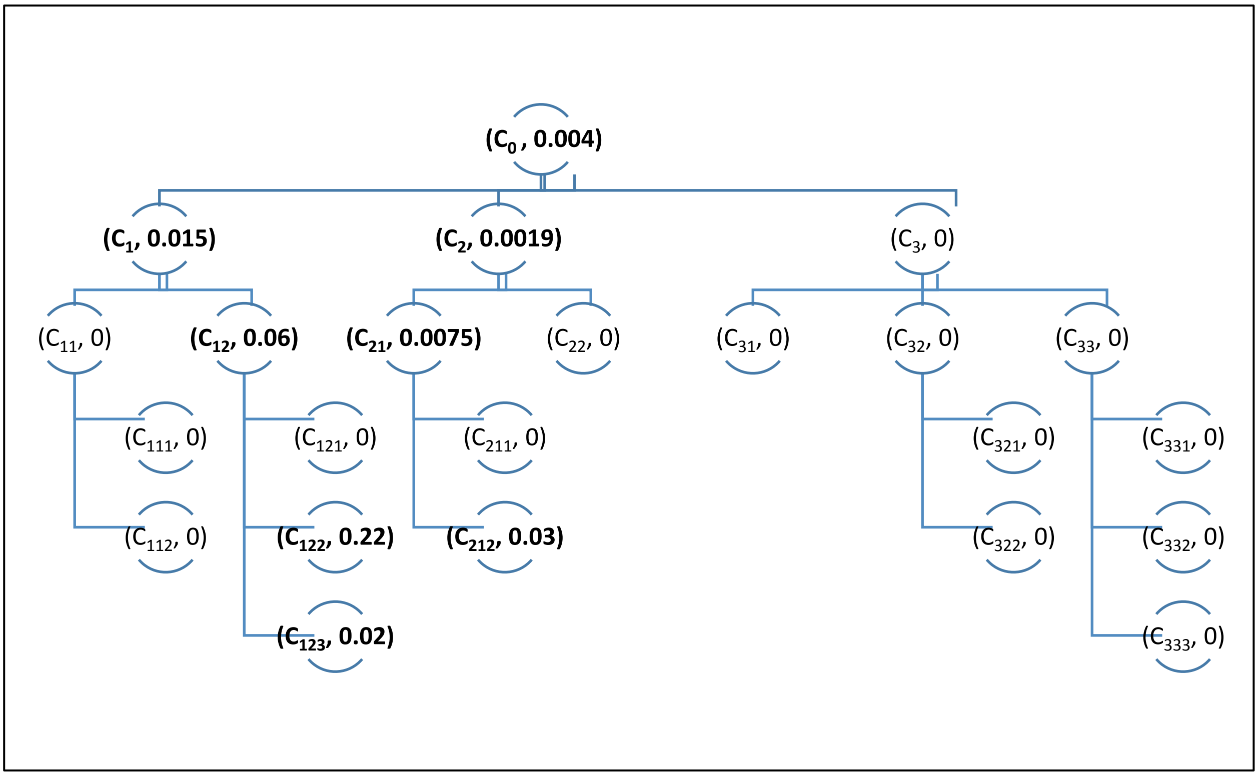

Figure 4.4. The user's tree after reading the paper P1.

Then, the weight value propagates to the grandfather node c_1 as follows:

$$SW_1 = \propto \times\ SW_{12} = 0.25 \times\ 0.06 = 0.015$$

The weight of concept c_{212} is updated to be $SW_{212} = 0.03$. Then, the weight of its direct parent c_{21} is updated as follow:

$$SW_{21} = \propto \times\ SW_{212} = 0.25 \times\ 0.03 = 0.0075$$

Then, the weight value propagates to the grandfather node c_2 as follows:

$$SW_2 = \propto \times\ SW_{21} = 0.25 \times\ 0.0075 = 0.0019$$

Then, the root's weight is updated as follows:

$$SW_0 = (\propto \times\ SW_1) + (\propto \times\ SW_2) = 0.25\ \times (\ 0.015 +\ 0.0019) = 0.004$$

Therefore, the resulted weights after the user read the first paper P_1 as shown in Figure 4.4 are:

$$SW_0 = 0.004$$

$$SW_1 = 0.015$$

$$SW_{12} = 0.06$$

$$SW_{122} = 0.22$$

$$SW_{123} = 0.02$$

$$SW_{2} = 0.0019$$

$$SW_{21} = 0.0075$$

$$SW_{212} = 0.03$$

Figure 4.5 shows the user's tree after reading the second paper P_2. The process of updating the concepts' weights is the same as the process that is explained for the first paper P_1. Figure 4.6 shows the user's tree after reading the third paper P_3. Finally, all the weights are divided by three to normalize the concepts' weights. The final normalized resulted weights after the user read the research papers P_1, P_2 and P_3 are presented in Figure 4.7.

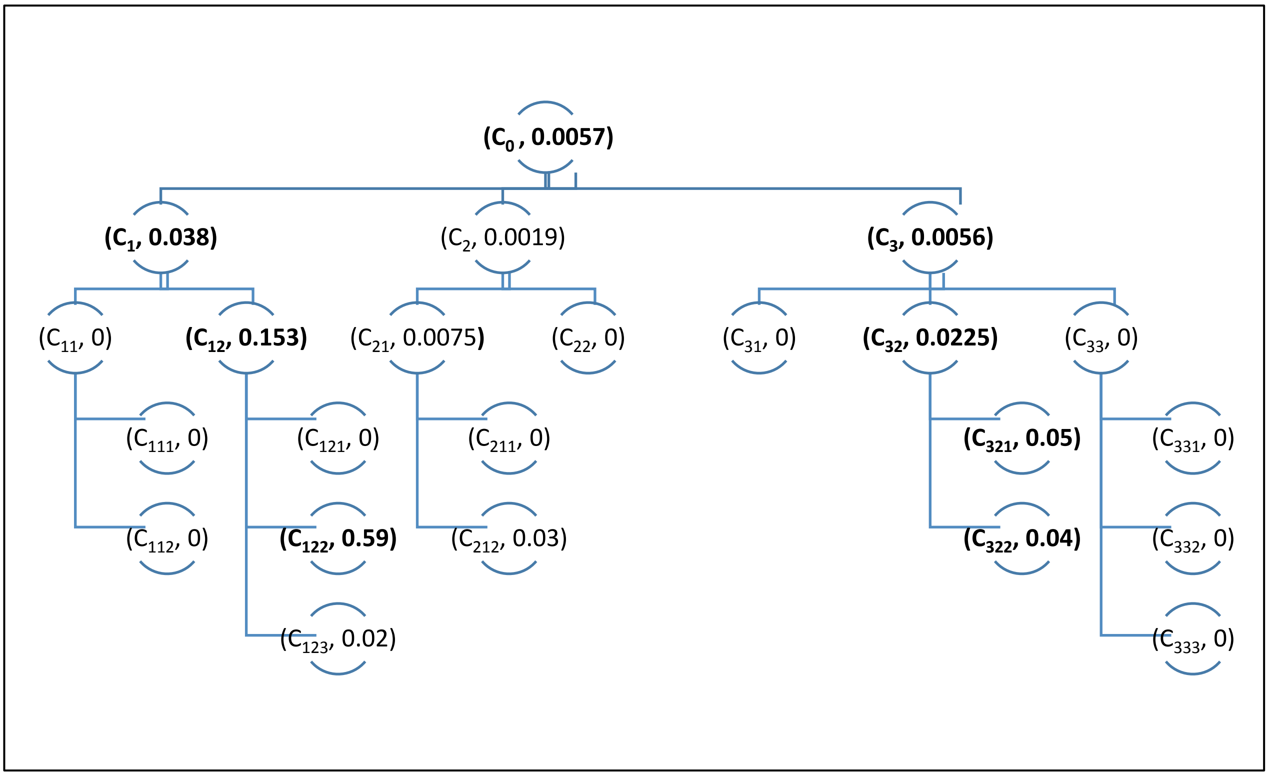

Figure 4.5. The user's tree after reading the paper P2.

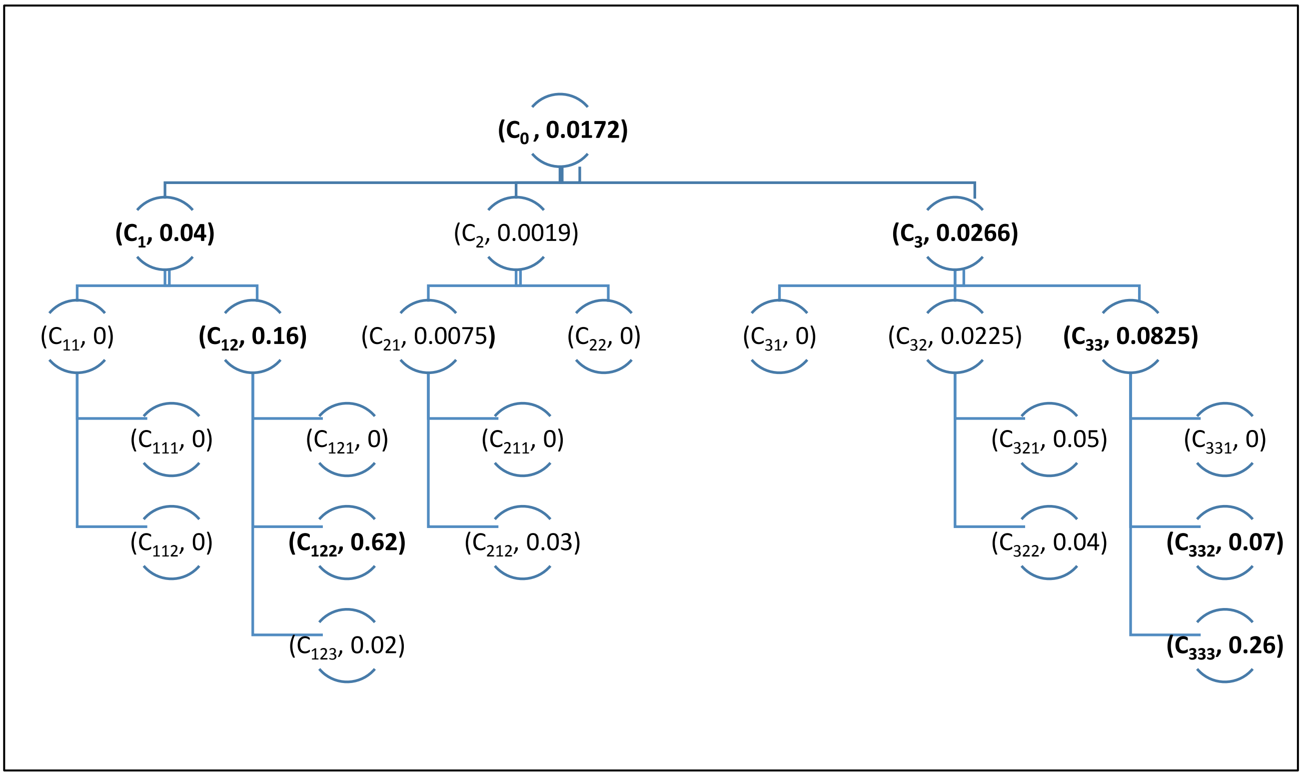

Figure 4.6. The user's tree after reading the paper P3.

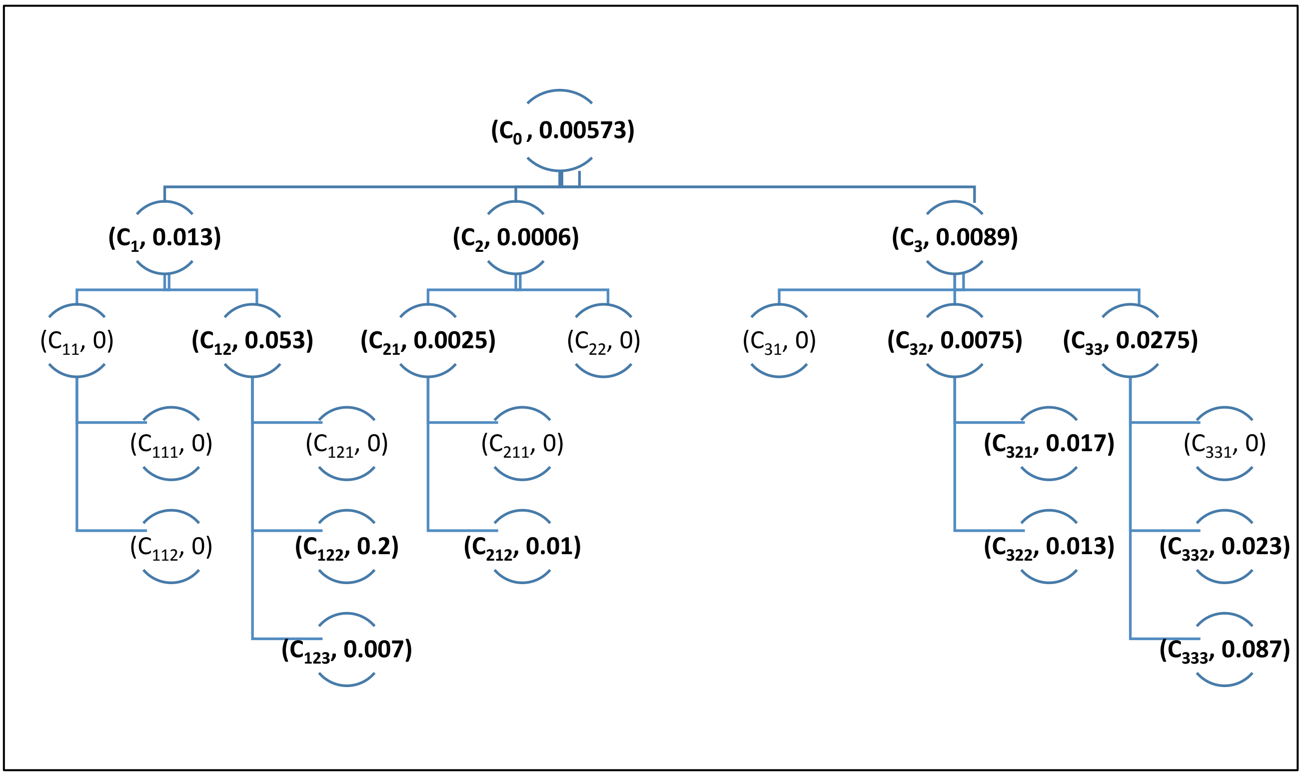

Figure 4.7. The final normalized user's tree after the user read all papers P1, P2 and P3.

4.4 Dynamic recommendation phase

In this phase, the trees of concepts for all research papers that the user has not read (unread research papers) are created. Then, the user profile, which is represented as a DNTC, is compared with the unread papers' trees of concepts to recommend the most relevant research papers to the user's interests. The details are as follows.

The outputs from the research papers classification phase and the DNTC user profiling phase are used as inputs to this phase. These inputs are: the papers' profiles and the user's DNTC profile. First, a tree of concepts is built for each unread paper in our dataset collection. A tree of concepts for an unread paper is built based on the top N concepts and their weights from the paper's profile, stored in the database which resulted from the research papers classification phase. The process for building the tree of concepts for a paper as follows. A tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts, the top N concepts and weights for this paper are retrieved from the profile database, and the weight values are propagated recursively to the parent nodes according to the equation (4.5). Figure 4.8 presents the algorithm to build the tree of concepts for the paper from the system's dataset collection to compare them with a user's tree of concepts.

Once the user profile and the research papers profiles are represented as trees of concepts, Tree Edit Distance (Lakkaraju et al. 2008) is used to calculate the distance between two trees (the user's tree and a tree of concepts for an unread paper). This distance is the cost of transforming one tree into another with the minimum number of operations. There are three types of operation: insertion, deletion and substitution. Insertion's cost is the cost of inserting a new concept into the tree with a given weight. Deletion's cost is the cost of deleting an existing concept with a given weight from the tree. Substitution's cost is the cost of changing a concept's weight to another weight. In our 2012 ACM CCS trees we suppose that the concept with zero weight is none existing node. Hence, the cost of deletion or insertion of a concept is equal to the weight associated with the concept, whereas the substitution cost is the difference between weights of an existing concept in both trees. Thus, the cost of modifying a

```
Build UnreadPapers Trees (UserID, OntologyTree, PapersProfiles, Alpha, TopN)
{
    Foreach UnreadPaper for UserID in dataset collection do
       {
         Get the TopN concepts and their corresponding weights from the Papers Profiles;
          Initiate a tree of concepts for a paper Pi;
          Foreach concept cj in the TopN concepts do
           {
             Find the concept cj in the paper Pi tree;
             Assign the concept cj weight = SWj;
             If the concept cj is not root do
               {
                 CurrentConcpet = cj;
                 CurrentConcept_SW = SWj;
                 Loop until the paper's Tree's root reached
                     {
                       get currentConcpet.Parent;
                       update currentConcpet.parent weight:SWP += CurrentConcept_SW * Alpha;
                       CurrentConcpet = currentConcpet.Parent;
                       CurrentConcept_SW = SWP;
                     }
               }
           }
       }

 }
```

Figure 4.8. The algorithm to build the papers trees of concepts.

tree of concepts for a paper to match the user tree is calculated. The two most similar trees are those which have the lower total cost of transformations between them. After calculating the total cost between all trees of concepts for the research papers and a user tree, the total cost with its associated id of the paper (PaperID) are stored a list and sorted in increasing order. Hence, the closest research papers to the user's preferences appear first and the most distant research papers appear last. The final output of the recommendation phase is a list of ordered recommended research papers.

In our system the Tree Edit Distance technique runs dynamically every time the user reads a new paper from the system dataset collection. Figure 4.9 presents the algorithm for our Dynamic Tree Edit Distance technique.

```
Dynamic Tree Edit Distance (UserTree, UnreadPapersTrees, CurrentTime)
{
    //m is the number of unread papers in CurrentTime.
    Create an array (ECosts [m]) to save the edit distance costs for each paper;
    Foreach UnreadPaperTree PTi do
       {
          W1=0, W2=0;
          Foreach concept cj in UserTree do
          {
              Get the concept cj weight in UserTree SWUj;
              Find the concept cj in UnreadPaperTree PTi and its weight SWPTij;
               W1 = SWUj;
               W2 = SWPTij;
               Absolute = |W1-W2|;
               Ecost [PTi] += Absolute;
          }
        }

    Sort the array Ecosts [m] in increasing order;
  }
```

Figure 4.9. Dynamic Tree Edit Distance technique algorithm.

4.5 Evaluations and results

In order to measure the performance of the proposed system, we evaluate:

1) The accuracy of the classifier model.
2) The performance of our DNTC user profiling and recommendation method.

For these purposes we introduce two evaluation experiments. The first experiment aims to evaluate the classification performance for mapping research papers in a dataset. The second experiment evaluates the performance of our DNTC recommendation method. The programming language that is used to create the classifier and the recommendation model is C# language. The database is created by using the Microsoft SQL server.

4.5.1 Evaluation for the classification phase

The ACM provided us with a dataset that contains 16,307 mapped research papers for the 2012 ACM CCS ontology. The main categories in the 2012 ACM CCS that are evaluated are: Hardware, Computer Systems Organization, Networks, Software and Its Engineering, Information System, Theory of Computation, Mathematics of Computing, Security and Privacy, Human-Centered Computing and Computing Methodologies. The total number of the concepts under these categories is 1,329 concepts, and the number of leaf concepts is 986 concepts. Table 4.1 shows a sample of the database for some concepts in the 2012 ACM CCS ontology.

The research papers are mapped by the authors of the research papers. The authors of the research papers are allowed to assign their research papers to more than one leaf concept. Title, abstract and authors' keywords are used from each paper in our classifier. To evaluate the accuracy of our classifier, 50% of the ACM dataset is used as training set and the other 50% as the testing set. The research papers from the training set used to learn a concept (c_j) are all combined into one training document file (d_j). All terms in this file are converted to vectors with their weights using the TF-IDF as explained in section 4.2.1.

Concept ID	Concept name	Concept's Training Document
1141	Data dictionaries	DataDictionaries.txt
1152	Data warehouses	DataWarehouses.txt
1161	Database web servers	DatabaseWebServers.txt
1162	Object-relational mapping facilities	ObjectRelationalMappingFacilities.txt
1163	Distributed transaction monitors	DistributedTransactionMonitors.txt
1211	Enterprise resource planning	EnterpriseResourcePlanning.txt
1221	Synchronous editors	SynchronousEditors.txt
2311	Local area networks	LocalAreaNetworks.txt
2331	Error detection and error correction	ErrorDetectionAndCorrection.txt
3121	Embedded software	EmbeddedSoftware.txt
3251	Design languages	DesignLanguages.txt

Table 4.1. A sample of the database for some concepts in 2012 ACM CCS.

Following this training phase, research papers in the testing set are classified as explained in section 4.2.2. The output for each paper is stored as the paper's profile. If the highest weighted concept resulted from the classifier is one of the concepts that are chosen by the paper's authors, then we consider it as positively classified. We evaluate the performance using the following equation:

$$Accuracy = \frac{Positive\ classified\ papers}{All\ papers} \tag{4.6}$$

Figure 4.10 shows the accuracy results for our classifier with 50% training set and 50% testing set for the main categories in the 2012 ACM CCS. The accuracy results may depend on the distribution of concepts in the training set. For example, the concepts with significant high accuracy result (92%) under Information systems and Human-centered computing may have good representation among the training research papers. The concepts with low accuracy results, such as (79%) under Networks, may have poor representation among the training research papers. The average of the classification results in accuracy for all categories is 86.5%.

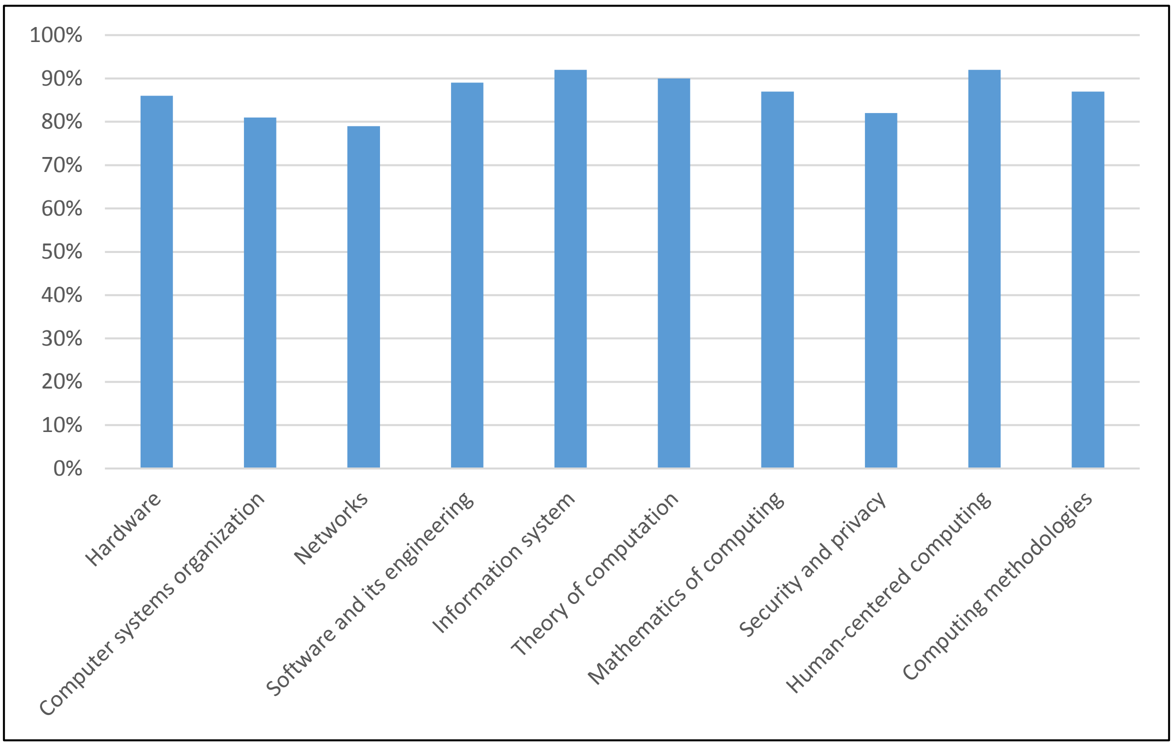

Figure 4.10. Accuracy of research papers classification phase with 50% training set and 50% testing set.

Table 4.2 shows the accuracy results for our classifier for different sizes of training/testing sets and with/without authors' keywords for the main ten categories in the 2012 ACM CCS. The accuracy results for each category may depend on the quality of the training set that is provided by ACM. We analysed the results with/without authors' keywords to realise the benefit from using authors' keywords in the classification process. It can be seen clearly that using authors' keywords provide better results in accuracy than without authors' keywords. For example, the average of accuracy for the classifier is 89.2% with authors' keywords by using 70% of the ACM dataset as the training set, whereas it is 81.8% without authors' keywords. When it comes to the size of training/testing sets, the last row in Table 4.2 shows that the average of the accuracy of the classifier is increased when the size of the training set is increased. For instance, the average accuracy is 86.5% (with authors' keywords) when the size of the training set is 50%, then it increases to 89.2% when the size of the training dataset is increased to be 70%. Therefore, we retrained the classifier using all the research papers in the ACM dataset as the training set to classify the research papers in two datasets: the CiteSeerX dataset and the BibSonomy dataset in next section to create the paper profile databases which serve as our datasets in the subsequent experiments.

Category	**With authors' keywords**			**Without authors' keywords**		
	Train/Test 50%/50%	Train/Test 60%/40%	Train/Test 70%/30%	Train/Test 50%/50%	Train/Test 60%/40%	Train/Test 70%/30%
Hardware	86%	88%	89%	74%	78%	80%
Computer systems organization	81%	83%	84%	72%	77%	81%
Networks	79%	82%	83%	69%	73%	77%
Software and its engineering	89%	91%	92%	76%	80%	83%
Information system	92%	94%	94%	78%	82%	85%
Theory of computation	90%	91%	92%	77%	80%	83%
Mathematics of computing	87%	89%	91%	73%	76%	80%
Security and privacy	82%	84%	84%	70%	74%	79%
Human-centered computing	92%	93%	93%	78%	83%	86%
Computing methodologies	87%	89%	90%	72%	79%	84%
Average of accuracy	86.5%	88.4%	**89.2%**	73.9%	78.2%	**81.8%**

Table 4.2. Accuracy of the research papers classification phase with different sizes of training/testing sets and with/without authors' keywords.

4.5.2 Evaluating the performance of the DNTC recommender system

The evaluation process of recommender system algorithms is known to be difficult and expensive as these systems are typically complex and have many components, properties and parameters which have to be examined in order to provide the optimum performance (Li et al, 2014). Therefore, offline evaluation methodology is used for our evaluation to measure the performance of the proposed DNTC recommender system. We have two approaches of offline evaluation:

1- Simulated users for the CiteSeerX dataset.

2- Real users' records from the BibSonomy dataset.

We used the simulation approach because when we finished from developing our DNTC model, we were searching for a rich dataset that contains real users' records. However, we were unable to get access to such dataset. Hence, we used research papers from the CiteSeerX dataset and we opted to use user behaviour simulation approach to test specific scenarios for multiple concepts and variant range of papers quantity. After we finished from the DNTC model and while we are developing the short-term and long-term models in the next chapter, we were able to get access to the BibSonomy dataset. Therefore, we revaluated our DNTC model using the BibSonomy dataset.

4.5.2.1 Evaluating our DNTC model with the CiteSeerX dataset

The CiteSeerX is a search engine and digital repository of scientific and research papers. It is a collection of over 5 million research papers primarily in the field of computer and information science. We used 100,000 research papers as a subset of that collection. This subset of CiteSeerX's research papers are entered to our classifier to classify them according to the 2012 ACM CCS ontology and we then use them as our dataset to evaluate the performance of our DNTC recommender system in this section.

We implemented the users' simulation using C# language and Microsoft SQL server. We created user scenarios that simulate users' interests and preferences for multiple concepts and variant range of research papers quantity. Nine main templates for user scenarios are created to simulate different numbers of concepts that represent multiple user interests. We have 3 main types of template scenarios that consider a different number of concepts during user's reading: three concepts, four concepts and five concepts as follows (see Appendix A):

a) Users' template scenarios 1, 2 and 3 consider **three** concepts.

b) Users' template scenarios 4, 5 and 6 consider **four** concepts.

c) Users' template scenarios 7, 8, and 9 consider **five** concepts.

Each type has three different scenarios that involve a different quantity of research papers during user's reading. There is small quantity (15 research papers in scenarios 1, 4 and 7), medium quantity (30 research papers in scenarios 2, 5 and 8) and large quantity (50 research papers in scenarios 3, 6 and 9). Each scenario template is applied on the ten main categories in the 2012 ACM CCS: Hardware, Computer systems organization, Networks, Software and its engineering, Information system, Theory of computation, Mathematics of computing, Security and privacy, Human-centered computing and Computing methodologies. Hence, we have 90 virtual users (i.e. 9 templates*10 main categories). The concepts are selected randomly from the main categories in the 2012 ACM CCS to create each individual user's scenarios using the templates. The research papers for each concept are chosen randomly from our classified CiteSeerX dataset that resulted from the classification phase.

4.5.2.1.1 Evaluating α and TopN parameters with the CiteSeerX dataset

In this section, we evaluated different values for α (the propagation factor) and *TopN* (the number of the top related concepts for a paper) parameters to find the optimal values that provide the best overall performance for our recommender system. The measurement that is used for evaluation is Mean Average Precision (MAP). The MAP for M users is the average of the average precision of each user (Manning et al., 2008):

$$MAP = \frac{\sum_{i=1}^{M} AVG\ P}{M} \qquad (4.7)$$

We calculated the average precision (AVG P) for each user as follows (Jannach et al., 2010):

$$AVG\ P = \frac{P_{10} + P_{20} + P_{30}}{3} \qquad (4.8)$$

Where P_{10}, P_{20} and P_{30} are precisions for cut-off results for top 10, 20 and 30 recommended research papers. The precision for cut-off results at position k (P_k) is used to evaluate the top k recommended research papers as follows (Jannach et al. 2010):

$$\boldsymbol{P_k} = \frac{\textbf{\textit{Number of relevant recommended papers to a user}}}{\boldsymbol{k}} \qquad (4.9)$$

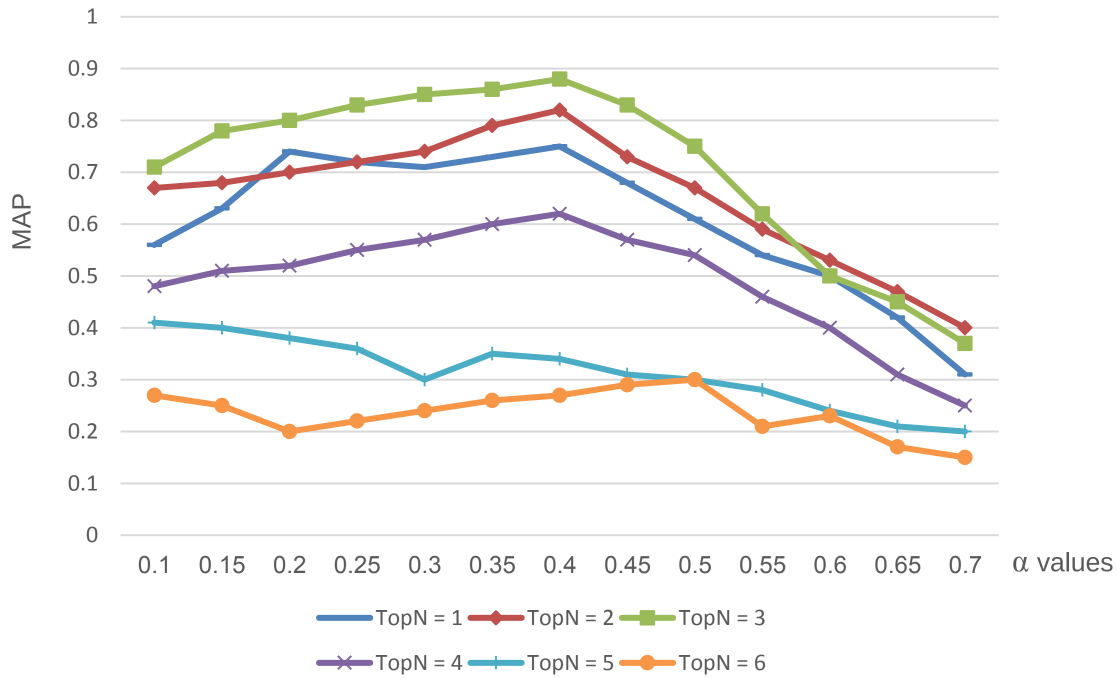

Figure 4.11. The MAP results using different α and TopN values with CiteSeerX.

Figure 4.11 shows the MAP results of applying our recommender system on all the users' scenarios using different α and *TopN* values. It can be clearly seen that the MAP results for *TopN*= 6 are relatively low. This is because the top 6 related concepts are a very large number of concepts to be included during build user and paper trees of concepts. The MAP results increase whenever the *TopN* value decreases until *TopN*=3. When *TopN*=3, we have the best results because the top 3 similar concepts to a paper might hold the most essential concepts that are expected to be related to this paper, while considering just the top 1 or 2 concepts may omit some of the very significant concepts.

We tested our system with different values for α in the range of [0.1 to 0.7]. Figure 4.11 shows that the MAP results improve when α value comes close to 0.4 and *TopN* values decrease, and clearly the MAP results tend to decrease when reaching the smallest or largest values (i.e. 0.1 and 0.7 respectively). The results are very low when $\alpha = 0.7$, because the propagation value is very large, and then large values are propagated over the reference ontology that makes recommending the correct interests is difficult. When $\alpha = 0.1$, most of the research papers were mapped to the leaf concepts from the reference ontology which make the recommendations to be too specific to represent all the users' interests. When α=0.4, the MAP results improve considerably as this value maintains a balance between general and specific concepts. According to these results, we assign α to be 0.4 and *TopN* to be 3 in our system with the CiteSeerX dataset.

4.5.2.1.2 Comparing our system against baselines with the CiteSeerX dataset

In this section, we compared the DNTC system against two baselines. Baseline 1 is recommender system using the dynamic vector of concepts (DVC) where there is no propagation of weights to parents (i.e. α=0). Baseline 2 is recommender system using the non-normalized tree of concepts (NNT) (Chandrasekaran et al., 2008).

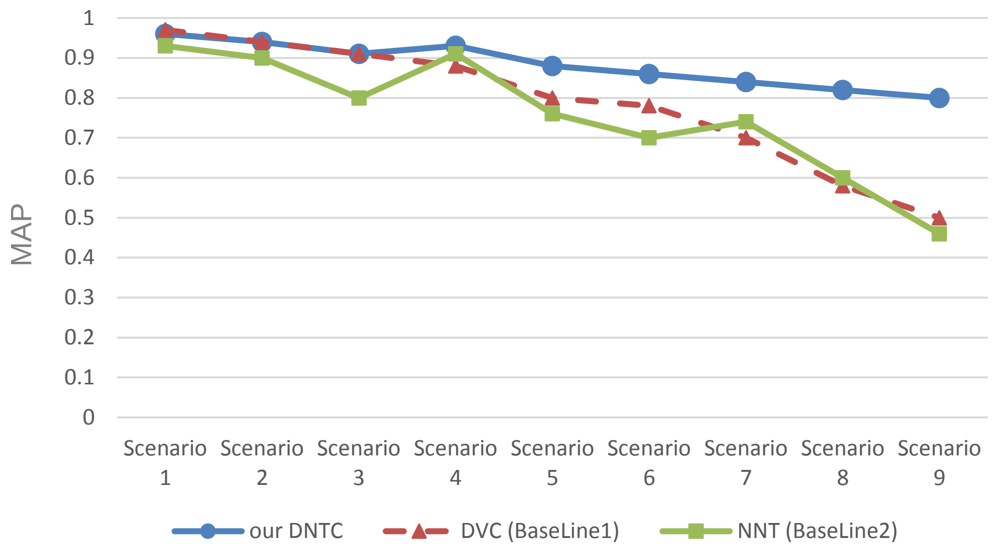

Figure 4.12. Comparing the MAP for each scenario with the three recommender systems.

Figure 4.12 shows the MAP for our DNTC system against the two baselines. For user scenarios 1, 2 and 3 that consider only three concepts, we can see that the results for the DVC system are comparable with our DNTC system. However, when the number of concepts is increased in the other scenarios to be more than three concepts, our DNTC system outperforms the DVC method. This is because with multiple concepts the task of user profiling and recommendation is more difficult for the recommender system based on vectors of concepts. For instance, scenarios 7, 8, and 9 consider five concepts during users' reading and there is a substantial improvement in the MAP for these scenarios by using our system. Therefore, when a user reads multiple concepts, our system based on the tree of concepts significantly outperforms the system that based on vectors of concepts.

When it comes to the NNT system, Figure 4.12 shows that when the quantity of research papers is small as in scenarios 1, 4 and 7 (that involve 15 research papers), the results for the NNT system are slightly lower than our system. However, when the quantity of research papers is increased to be 30 research papers with more than three concepts in scenarios 5 and 8, the results for the NNT system decline significantly compared with our system. The NNT system's results dramatically drop when the number of research papers becomes 50 research papers in scenarios 3, 6 and 9. This is because the NNT system does not normalize the concepts' weights in the user's tree of concepts to be appropriate to compare them with the concepts' weights in a paper's tree of concepts. Hence as the user reads more research papers, the weights in the user profile grow and become less and less comparable with the weights in the profile of a single paper.

Finally, both the DVC and the NNT systems achieved the lowest performance in Figure 4.12 at scenario 9, where the scenario considers five concepts of interests and 50 research papers. The MAP at scenario 9 for the DVC system is 0.5 and for the NNT system is 0.46, whereas the MAP for our DNTC system is 0.8. The DNTC system did not drop dramatically as the DVC and the NNT systems. Therefore, when a user reads multiple concepts and a large quantity of research papers, our system significantly outperforms both of the baseline systems. Table 4.3 shows the MAP for all users' scenarios that reflect the results of those of Figure 4.12. Our DNTC system has the highest MAP of 0.88, while the DVC system scored the second best MAP (i.e. 0.78) while the NNT system achieved the lowest MAP of 0.76 (These results are published in Al Alshaikh et al., 2017a).

Recommender system	**$MAP_{CiteSeerX}$**
Our system (DNTC)	0.88
Baseline 1 (DVC)	0.78
Baseline 2 (NNT)	0.76

Table 4.3. The MAP results for the three systems with the CiteSeerX dataset.

4.5.2.2 Evaluating our DNTC model with the BibSonomy dataset

The BibSonomy dataset contains actual records of users' interests as posts for research papers over approximately a ten year period. Each post contains: metadata for a research paper, date and time of the post. The posts are ordered in chronological order. These posts are considered as users' reading records of research papers. From the metadata we extracted a paper's title, abstract and keywords. Then, each paper is entered to our classifier to classify it according to the 2012 ACM CCS ontology. We used records of users reading behaviour over the years 2015 and 2016 for the users in the field of computer and information science. This includes 1,642 users and 43,140 research papers. The users with fewer than 50 research papers are removed, hence the remaining users are 1,201. We used those users' records to evaluate the performance of our DNTC recommender system in this section. Table 4.4 shows a sample of the database for the users' records from the BibSonomy dataset. Table 4.5 illustrates a sample of the extracted metadata for the papers. The measurement that is used for evaluation is Mean Average Precision (MAP) as in section 4.5.2.1.1. The MAP is evaluated for all users every time a new paper read by the users.

User ID	Paper ID	Post date and time
683914	23031526	2015-05-29 13:43:20
950338	23036081	2015-05-30 14:57:23
647445	23036873	2015-06-01 15:48:15
683914	23036972	2015-06-02 10:51:25
683914	23036973	2015-06-02 12:42:12
500721	23036946	2015-06-02 13:25:32
950338	23036897	2015-06-03 15:34:11

Table 4.4. A sample of the database for users reading records in the BibSonomy dataset.

Paper ID	Title	Abstract	Keywords
23031526	The Akamai Network: A Platform for High-Performance Internet Applications.	Comprising more than 61,000 servers located across nearly 1,00 networks in 70 countries worldwide, the Akamai platform ……	Akamai, CDN, overlay networks, HTTP, DNS, content delivery, quality of service, streaming media.
23036081	EaCRS: an extendible array based compression scheme for high dimensional data.	Multidimensional arrays are becoming important data structure for handling large scale multidimensional data; e.g., in scientific databases or MOLAP databases. Due to the …….	Extendible Array, Database compression, Multidimensional Array, Compression ratio, MOLAP.
23036873	Functional programs as compressed data.	We propose an application of programming language techniques to lossless data compression, where tree data are compressed as functional programs that generate them. This "functional programs as compressed data"…….	Semantics based program manipulation Program transformation Data compression Functional programs Higher-order Model Checking
23036972	Spectrum-Efficient and Scalable Elastic Optical Path Network: Architecture, Benefits, and Enabling Technologies.	The sustained growth of data traffic volume calls for an introduction of an efficient and scalable transport platform for links of 100 Gb/s and beyond in the future optical …..	Optical fiber networks, Optical packet switching, Optical buffering, SONET, Synchronous digital hierarchy.
23036973	Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing.	Cloud computing offers utility-oriented IT services to users worldwide. Based on a pay-as-you-go model, it enables hosting of pervasive applications from consumer, scientific, and business domains. However,……	Energy efficiency Green IT, Cloud computing, Resource management, Virtualization, Dynamic consolidation.
23036946	Algorithmics on SLP-compressed strings: A survey.	Results on algorithmic problems on strings that are given in a compressed form via straight-line programs are surveyed. A straight-line program is a context-free grammar…..	Algorithms for compressed strings, compressed word problems, computational complexity.
23036897	LEaCRS: An Extendible Array Based Compression Scheme for High Dimensional Data Using Linearization.	Large multidimensional arrays are extensively used as the basic data structure in scientific, statistical and engineering applications. Increasing…..	MOLAP, Extendible Array, Compression Ratio, Array linearization function.

Table 4.5. A sample of the papers metadata.

4.5.2.2.1 Evaluating α and TopN parameters with the BibSonomy dataset

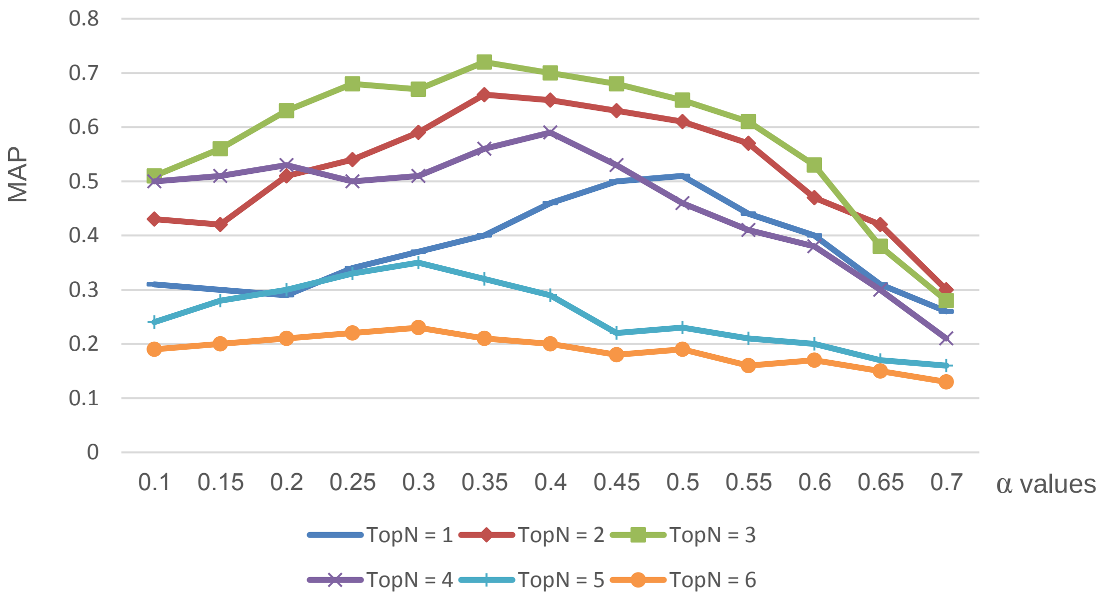

Figure 4.13. The MAP results using different α and TopN values with the BibSonomy dataset.

Figure 4.13 shows the MAP results of applying our DNTC system using different α and *TopN* values with the BibSonomy dataset. In general, the results with the BibSonomy dataset are lower than the results with the CiteSeerX dataset in Figure 4.11 (i.e. the optimal result with the BibSonomy dataset in Figure 4.13 is 0.72, whereas with the CiteSeerX dataset in Figure 4.11 is 0.88). This is because real users' behaviour is more complex than simulated users. Nonetheless, the pattern of the results using different α and *TopN* values are similar in both datasets the CiteSeerX and the BibSonomy. We can see that the lowest results are for *TopN*= 6 and *TopN*= 5. This is because the top 6 and top 5 related concepts are large numbers of concepts to be included during build the user and the paper trees of concepts. The best results are when *TopN*=3, while considering just the top 1 or 2 concepts may omit some of the very significant concepts. The α value is tested in the range of [0.1 to 0.7]. It can be seen that the MAP results with *TopN*=2 and *TopN*=3 improve when α value comes close to 0.35, as this value maintains a good equilibrium between general and specific concepts for complex real users' behaviour. The optimal result is when *TopN*= 3 and α = 0.35. According to these results, we assign *TopN* = 3 and α = 0.35 in our system with the BibSonomy dataset.

4.5.2.2.2 Comparing our systems against baselines with the BibSonomy dataset

We compared our DNTC system against the same two baselines in section 4.5.2.1.2: baseline 1 (DVC) and baseline 2 (NNT). Figure 4.14 presents the MAP for our DNTC system against the two baselines. Every time the users read a new research paper, the MAP of all user is calculated for the three systems until 50 research papers are read by the users. Every time a user reads a new research paper typically means a larger quantity of research papers and includes a larger quantity of interesting concepts. At the beginning of Figure 4.14, all the three systems started to learn the user profile. Our DNTC system speedily learnt the users' interesting concepts. However, after the 8th paper the users' behaviour became more intricate. Until the 8th paper, the NNT system performance is slightly lower than our DNTC system. The DVC system has the lowest performance because the quantities of concepts in the real users' records in the BibSonomy dataset are larger than three concepts (more details and analyses about concepts quantity in section 5.1.2). The real users' records in the BibSonomy show that the research papers that are read by a user can involve a large distribution of concepts. Therefore, the performance of the DVC system declined from the beginning of the experiment. This is because with multiple concepts the task of user profiling and recommendation is more difficult for the recommender system based on vectors of concepts. After the 10th paper, the performance of the NNT system declined dramatically because it does not normalize the concepts' weights in the user's tree of concepts to be appropriate to compare them with the concepts' weights in a paper's tree of concepts. Hence as the user reads more research papers, the weights in the user profile grow and become less and less comparable with the weights in the profile of a single paper.

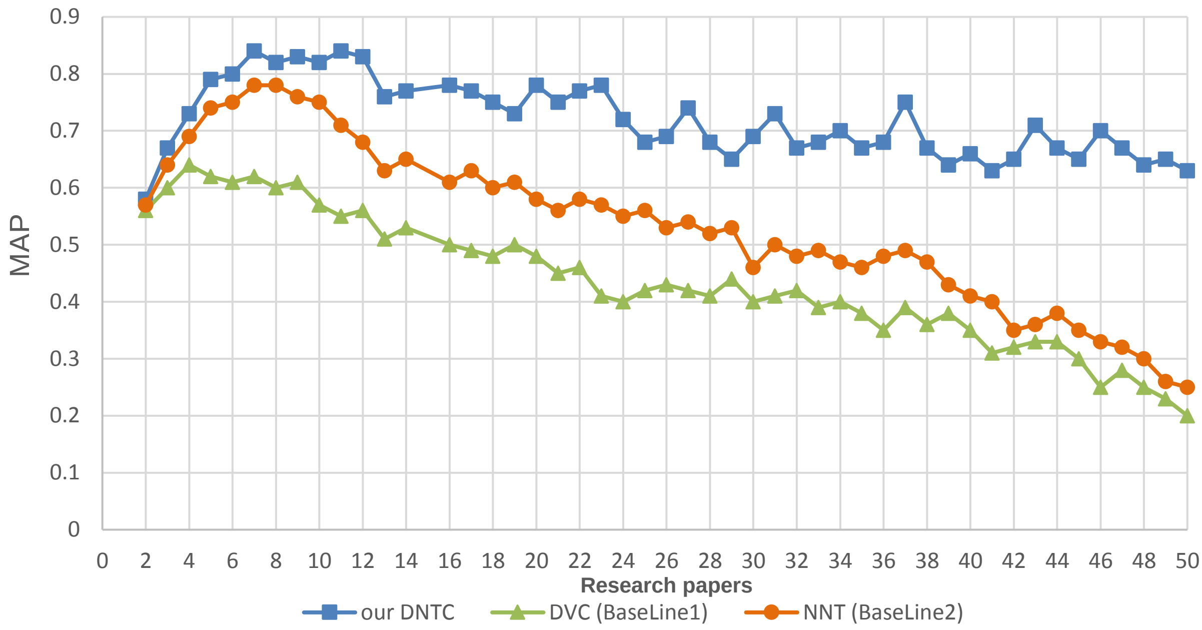

Figure 4.14. Comparing MAP for the three recommender systems with BibSonomy.

At the end of the experiment with 50 research papers, both the DVC and the NNT systems achieved the lowest performance in Figure 4.14 (i.e. 0.2 and 0.25 respectively), where the users' records involved a large quantity of research papers and a large quantity of concepts. The third column in Table 4.6 shows the average of the MAP results in Figure 4.14 with the BibSonomy dataset. The DVC system achieved the lowest performance with the MAP of 0.44, then the NNT system achieved the second lowest performance with the MAP of 0.53, whereas our DNTC system achieved the best performance with the MAP of 0.72.

From Table 4.6 we can compare the results between the simulated users approach with the CiteSeerX dataset and the real users' records approach with the BibSonomy dataset. It can be seen clearly that the three systems' results for the BibSonomy are lower than the CiteSeerX' results. This is because the real users' behaviour is more complex than the simulated users. However, our DNTC system achieved the highest MAP results for both approaches. For the CiteSeerX dataset, the DNTC system has 10% higher MAP result than the DVC system and 12% higher than the NNT system. For the BibSonomy dataset, the DNTC system has 28% higher MAP result than the DVC system and 19% higher than the NNT system. These results demonstrate that the proposed DNTC system effectively outperforms the other two systems with real complex users' behaviour and is able to provide high average precision when a user has multiple concepts and read a large quantity of research papers.

Recommender system	$MAP_{CiteSeerX}$	$MAP_{BibSonomy}$
Our system (DNTC)	0.88	0.72
Baseline 1 (DVC)	0.78	0.44
Baseline 2 (NNT)	0.76	0.53

Table 4.6. The MAP results for the three systems with two datasets.

4.6 Conclusions

In this chapter, we presented a novel recommender system for research papers which used a Dynamic Normalized Tree of Concepts (DNTC) as the user modelling technique. The DNTC system utilizes the ontology for the 2012 ACM CCS, which is far richer and more complex than the previous 1998 ACM CCS ontology. The user profiling phase creates a user profile as a dynamic normalized tree of concepts which is used with a dynamic tree edit distance method to compare between the user profile and the new unseen research papers that are also represented as a tree of concepts. We performed two approaches of offline evaluations to evaluate the performance of the proposed system. The first approach is simulated users with the CiteSeerX dataset. The second approach is the real users' records with the BibSonomy dataset. We compared our DNTC system against two baselines: recommender system using the dynamic vector of concepts (DVC) and recommender system using the non-normalized tree of concepts (NNT). Our results show that our novel DNTC model significantly outperforms both the DVC and the NNT systems in both approaches. We found that the simulation approach can indicate the performance of a system in case the real users' records are not available. Nonetheless, the DNTC system with the real users' records, in the BibSonomy dataset, provides significantly better improvements in the recommendations than the DVC and the NNT comparing with the simulation approach with the CiteSeerX dataset. This is because the real users' behaviour is further complex than the simulated users. Therefore, the DVC and the NNT systems are not able to handle the complexity of the users' behaviour. With the BibSonomy dataset, the DNTC system has 28% higher MAP result than the DVC system and 19% higher than the NNT system. Therefore, we can conclude that our novel DNTC system is able to provide high average precision when a user read a large quantity of research papers and has a large distribution of multiple concepts. In the next chapter, we will improve the DNTC system to be able to determine multiple concepts reflecting user's long-term and short-term interests.

Chapter 5. Novel Short-term and Long-term User Modelling Techniques for a Research Paper Recommender System

A major challenge in recommender systems is the modelling of dynamically evolving short-term and long-term user interests. The short-term interests represent the user's most recent interests which are more erratic, whereas the long-term interests are more stable in comparison (Challam et al., 2007). Recommender systems for research papers suffer from many limitations; for example, fast deviations in short-term interests may remain undetected and stable long-term interests may not be appropriately updated to reflect the user's evolving short-term and long-term interests. The importance of this stems from the need to design automatically adaptable user profiling techniques that should keep track of multiple information that is needed by the user. Therefore, there is a need for user profiling models and techniques that automatically adapt to the diverse and frequently changing users' short-term and long-term interests. We aim in this chapter to improve our DNTC model to adapt to the user needs for multiple concepts during his/her short and long term goals. The objective of this is to consider multiple user interests and develop novel mechanisms that allow the user profile to inject any new coming concepts and forget the no longer interesting concepts. Existing short-term and long-term user modelling techniques have been developed for domains such as recommending web pages (Gao et al., 2013, Hawalah and Fasli, 2015 and Li et al.,2007) and news articles (Zeb and Fasli, 2011, Agarwal and Singhal, 2014 and Zeb and Fasli, 2012), where a user reading behaviour is different from the research paper domain. These models depend on continuous time-based user behaviour measured in days for the web pages domain and in hours in the news domain. These models also assume that users are continuously active in their reading with no significant breaks. In this chapter, first we present the analysis of users' reading behaviour with research papers using the BibSonomy dataset. Then, we propose novel user modelling methods for short-term and long-term interests. The *short-term model* is based on a novel personalized dynamic sliding window (PDSW) technique where the window length is adapted according to the ratio between the number of concepts and number of research paper recently read by the user. The contents of these research paper are then used to build the user's short-term profile. The *long-term model* determines the user's long-term concepts and then selects the research papers that represent those concepts. The user's long-term profile is built from the selected research papers. The rest of this chapter is organized as follows. Section 5.1 presents the analysis for users' reading behaviour for research papers using the BibSonomy dataset. Section 5.2 illustrates our short-term and long-term models. The evaluations and results that are produced by our models are discussed in section 5.3. Finally, the conclusions are presented in section 5.4.

5.1 Analysing users' reading behaviour of research papers

We used real users' records from the BibSonomy dataset over the years 2015 and 2016 for users in the field of computer and information science. This includes 1,642 users and 43,140 research papers. Our analysis involved automatically searching for patterns of users reading behaviour. Firstly, we analysed the periods of days and months that a user was inactive (an inactive day/month is a day/month that the user did not read any research paper). Secondly, we analysed the users' reading behaviour during the active months.

5.1.1 Analysis results for inactive periods of days and months

We analysed the periods of days and months that a user was inactive as follows:

a. Average number of consecutive inactive days during an active month. (An inactive day is a day that the user did not read any research paper.)

b. Average number of consecutive inactive months per year. (An inactive month is a month that the user did not read any research paper.)

Figure 5.1 shows the average number of consecutive inactive days in an active month. It can be seen that users are not active every day; they do not read research papers continuously. Also, users have different patterns of this short-term inactivity. For example, 9% of users are inactive for eight continuous days per active reading month. Therefore, using a fixed duration in time-based models for short-term user profiling is not suitable for this domain. This is because the users can be inactive for several days, which will lead to inaccuracies if modelled based on fixed time periods. Figure 5.2

presents the average consecutive inactive months per year. Our results show that users may not read for several months and may have long inactive periods. For example, our results show that 21% of users are inactive in reading research papers for three continuous months.

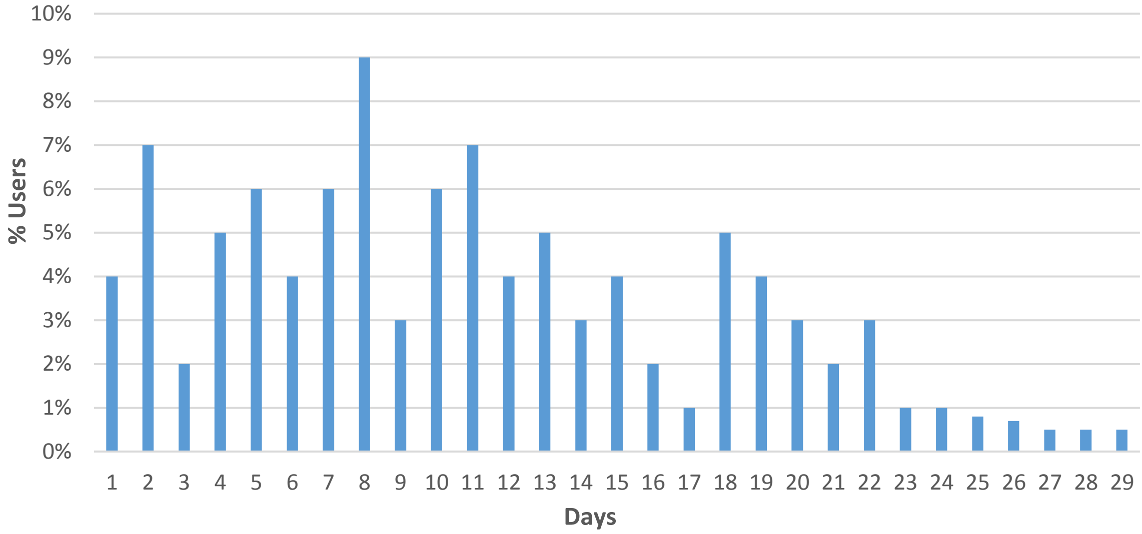

Figure 5.1. Average inactive days in an active month.

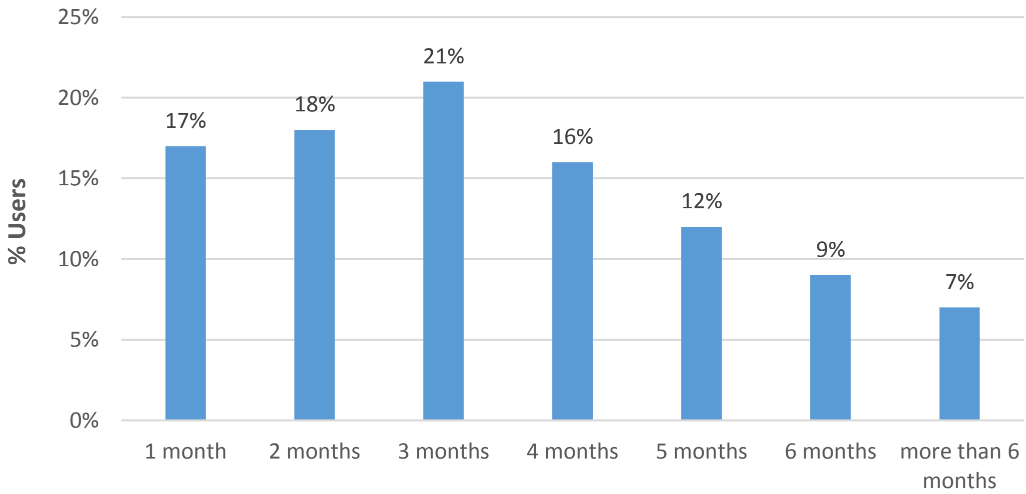

Figure 5.2. Average inactive months per year.

5.1.2 Analysis results for the users' reading behaviour during the active months

Our analysis for the users' behaviour during the active months includes the following:

a. Average number of research papers that are read by a user per active month.

b. Average number of concepts encountered in a user's reading per active month.

c. Number of long-term concepts that stay in a user's record more than one active month.

Figure 5.3 shows the average number of research papers read by a user per active month. There is significant variability in the number of research papers read by the users in an active month. For instance, 28% of the users read 6 to10 research papers and 23% of the users read 11 to 15 research papers per one active month.

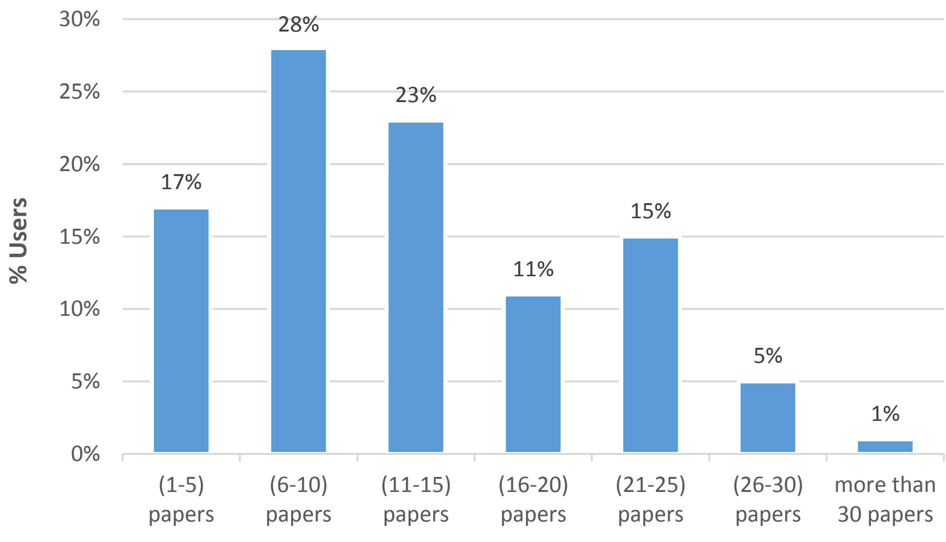

Figure 5.3. Average number of research paper per active month.

To analyse average number of concepts per active month, we used our classifier in section 4.2. Each paper in the BibSonomy dataset is classified to the top three most closely related concepts in the 2012 ACM CCS. Figure 5.4 shows the average number of concepts that are encountered by a user per active month. Figure 5.5 presents the number of long-term concepts that remain in a user's record for more than one active month. It can be seen that the number of long-term concepts in Figure 5.5 is fewer than the number of concepts in Figure 5.4. For example, the largest group of users in Figure 5.4 (34%) encounters 11-20 concepts per month, whereas the largest group of users in Figure 5.5 (28%) have 6-10 concepts remaining for more than one active month. This is because some of the concepts represented in Figure 5.4 can be short-term interests. Not all the short-term concepts can be considered as being long-term concepts. The current recommender systems for research papers do not involve short-term and long-term models; they mostly use the whole user reading history. Hence, they are not efficient in recommending the right research paper at the right time for evolving users' interests.

In general, our analysis shows that users are active during some days and inactive on other days. They may also be inactive for several months. Moreover, the users have different reading behaviours from each other, and reading behaviour for a user may change during a year. Therefore, utilizing continuous time-based models for building a user's profile based on continuous timing algorithms (such as Hawalah and Fasli, 2015) or time-based window (such as Gao et al., 2013) are not appropriate. Therefore, it is important to develop short-term and long-term models for a research paper recommender system. The next section presents our novel short-term and long-term models.

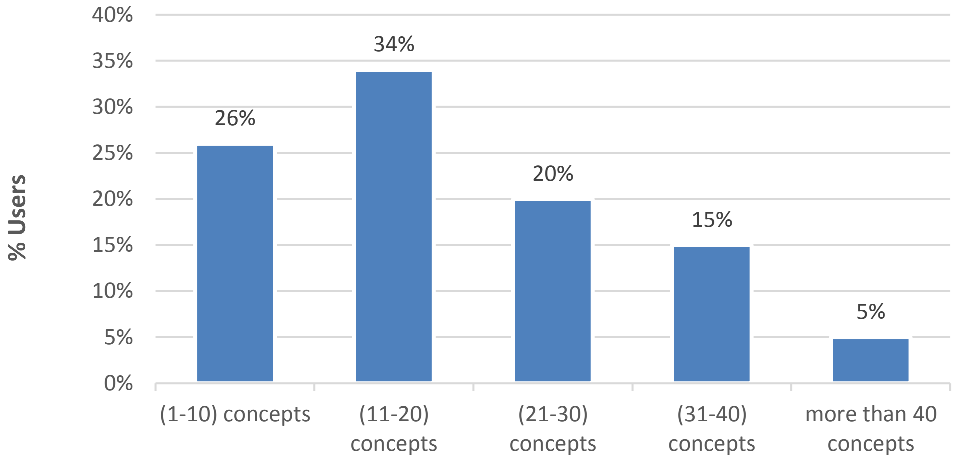

Figure 5.4. Average number of concepts per active month.

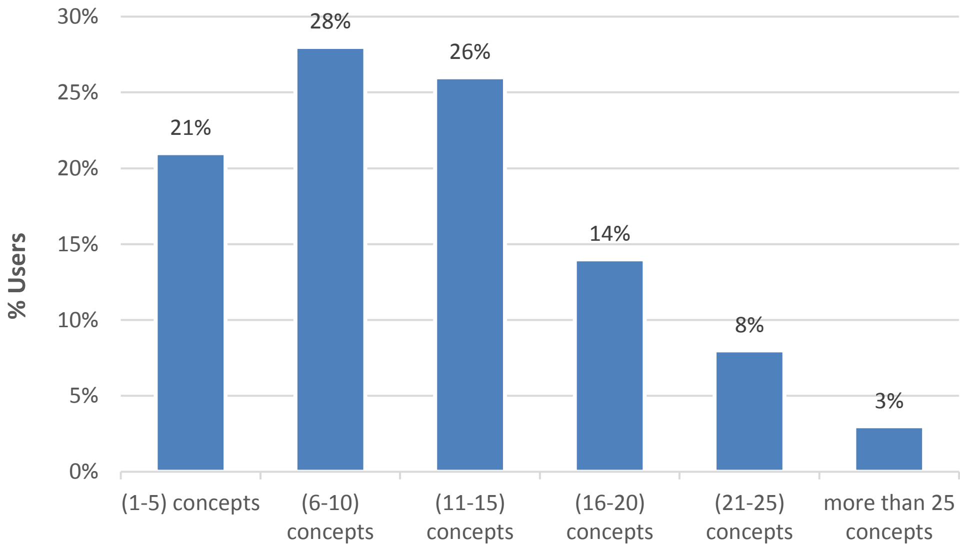

Figure 5.5: Number of long-term concepts.

5.2 Short-term and long-term models

In this section, we present our novel short-term and long-term models which automatically adapt to different users' reading behaviour. First the short-term model is described, then the long-term model is presented.

5.2.1 Short-term model

To improve our DNTC model to adapt to the user needs for multiple concepts during his/her short-term goals, the proposed short-term model uses our novel personalized dynamic sliding window (PDSW) technique. The PDSW length is the number of latest research papers that are read by a user. These research papers are then used to build a short-term user's profile, which is represented as DNTC profile as in section 4.3. Figure 5.6 presents the basic idea of the conceptual nature of the sliding window for the short-term model. In Figure 5.6 the PDSW length is four research paper. P_1 is the first paper read by the user, P_2 is the second paper and so on, the current time is T and the short-term user's DNTC tree is U_T. At time T, the sliding window is around the four papers P_1, P_2, P_3 and P_4, these papers are used to build the user's profile (U_T) as a dynamic normalized tree of concepts (DNTC) model. Then, at time T+1, the sliding window is moved to select the latest four papers (i.e. P_2, P_3, P_4 and P_5) to build the updated user profile U_{T+1}.

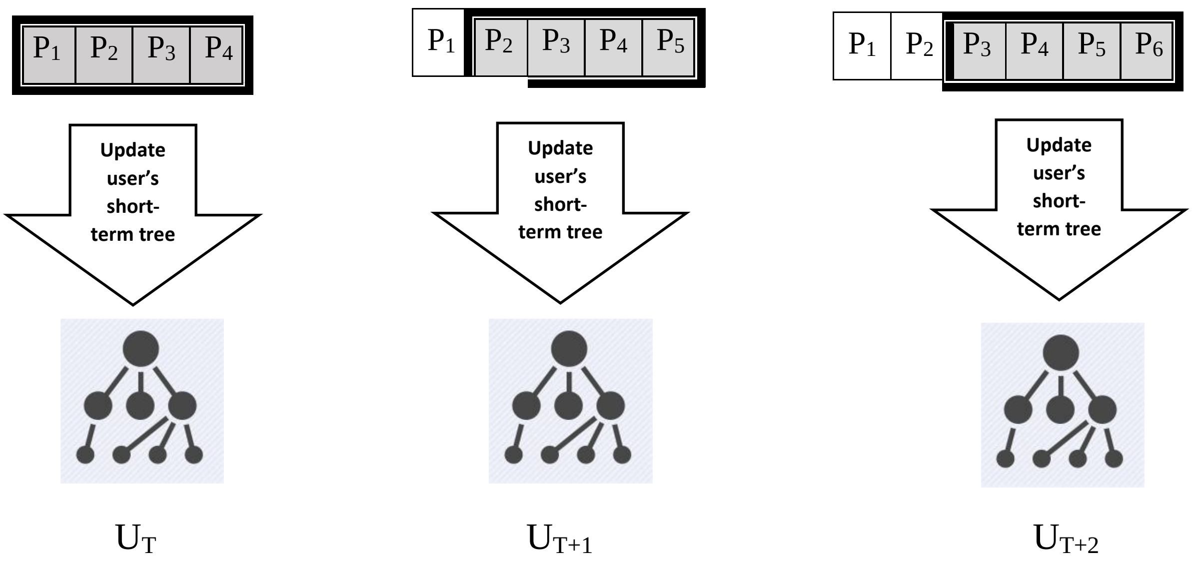

Figure 5.6. Building DNTC using our short-term sliding window.

The PDSW length is modified according to the ratio between the number of concepts and the number of research papers that are read by the user. The ratio R on time T is calculated for the previous active reading days for a user as follows:

$$R_T = \frac{\sum_{i=1}^{PAD_T} \frac{nC_i}{nP_i}}{PAD_T} \qquad (5.1)$$

where PAD_T is the number of previous active days on time T, nCi is the number of concepts in the active day i and nPi is the number of research papers in active day i. This ratio can indicate the changing in the user's behaviour. Therefore, the length of the sliding window is extended or shrunk according to the user's behaviour. Each time a new paper is read by a user, the new ratio R_{T+1} is compared with the previous ratio R_T. If R_{T+1} is larger than the R_T, then the previous PDSW length has a greater distribution of concepts and we have to shrink the PDSW length to focus on the latest research paper and concepts to discover the new short-term interests. If R_{T+1} is smaller than R_T, then we have to extend the PDSW length. If $R_{T+1}$is equal to the R_T then the window length remains unchanged. To shrink or extend the length (L) of PDSW, Signum function (sgn) is used as follows:

$$L_{T+1} = L_T + \beta * sgn\,(R_T - R_{T+1}) * R_{T+1} \qquad (5.2)$$

Where L_{T+1} is the new window length on time $T+1$, L_T is the previous window length on time T, β is decay factor and sgn function as follows:

$$sgn\,(R_T - R_{T+1}) = \begin{cases} -1 \; if \; R_T - R_{T+1} < 0 \\ 1 \; if \; R_T - R_{T+1} > 0 \\ 0 \; if \; R_T - R_{T+1} = 0 \end{cases} \qquad (5.3)$$

After calculating the new PDSW length, the latest research papers that are read by the user are selected to represent the user's short-term profile. The number of selected research papers is a rounding integer of the PDSW length (L_{T+1}). Then, the short-term user's profile is represented as DNTC profile as in section 4.3. Dynamic Tree Edit Distance technique as in section 4.4 is then used to recommend a set of research papers to the user that match his/her short-term interests.

Table 5.1 shows an example of the PDSW length during 20 days for a user. The value of the decay factor β in this example is 0.6. The initialization length (L_T) of the PDSW in the first day is equal to the number of the papers (*nPi*) read by the user in the first day. We can see that the PDSW window length L_T is extending or shrinking according to the changing in the user's behaviour. Then, the number of selected research papers in the last column in Table 5.1 is a rounding integer of L_T. Therefore, the number of the selected papers that are included in the short-term user's profile is increased or decreased based on the PDSW window length.

Active day i	***nPi***	***nCi***	***nCi/nPi***	***R_T***	***L_T***	***The number of selected research papers***
1	2	6	3.00	3.00	2.00	2
2	3	4	1.33	2.17	3.30	3
3	7	14	2.00	2.11	4.57	5
4	1	3	3.00	2.33	3.17	3
5	4	6	1.50	2.17	4.47	4
6	1	3	3.00	2.31	3.08	3
7	2	5	2.50	2.33	3.00	3
8	7	9	1.29	2.20	4.32	4
9	3	4	1.33	2.11	5.58	6
10	1	3	3.00	2.20	4.27	4
11	6	10	1.67	2.15	5.56	6
12	3	4	1.33	2.08	6.80	7
13	1	3	3.00	2.15	5.51	6
14	7	10	1.43	2.10	6.77	7
15	2	5	2.50	2.13	5.50	6
16	1	3	3.00	2.18	4.19	4
17	5	7	1.40	2.13	5.47	6
18	3	4	1.33	2.09	6.72	7
19	1	3	3.00	2.14	5.44	5
20	6	9	1.50	2.11	6.70	7

Table 5.1. An example of the PDSW length.

5.2.2 Long-term model

The long-term model is updated at the end of each active month for a user. Long-term concepts are the concepts that remain for more than one active month in a user's record. The long-term model selects the research papers that represent long-term concepts, then these research papers represent a user's long-term profile. The set of long-term concepts is defined as $LC = \{Lc_1, Lc_2,.., Lc_n\}$, where n is the total number of long-term concepts. After selecting the long-term concepts, the research papers that are related to at least one of the long-term concepts are selected to represent a user's long-term profile. The set of long-term research papers is defined as $LP = \{Lp_1, Lp_2,.., Lp_m\}$, where m is the total number of the long-term research papers and Lp_i is related at least to one of LC concepts. Then the set of research papers LP is used to build a user's long-term DNTC as in section 4.3. Figure 5.7 shows an example of the long-term model. There are ten research papers that are read by a user and four long-term concepts. There are only five research papers that are considered as long-term research papers. Paper2 and paper4 are related to concept Lc1. Paper5 is related to concept Lc2. Paper6 is related to two concepts Lc3 and Lc4. Paper9 is related to concept Lc4. These five research papers are used to build the user's long-term DNTC profile. Then, the Dynamic Tree Edit Distance technique as in section 4.4 is used to recommend a set of research papers to the user that match his/her long-term interests.

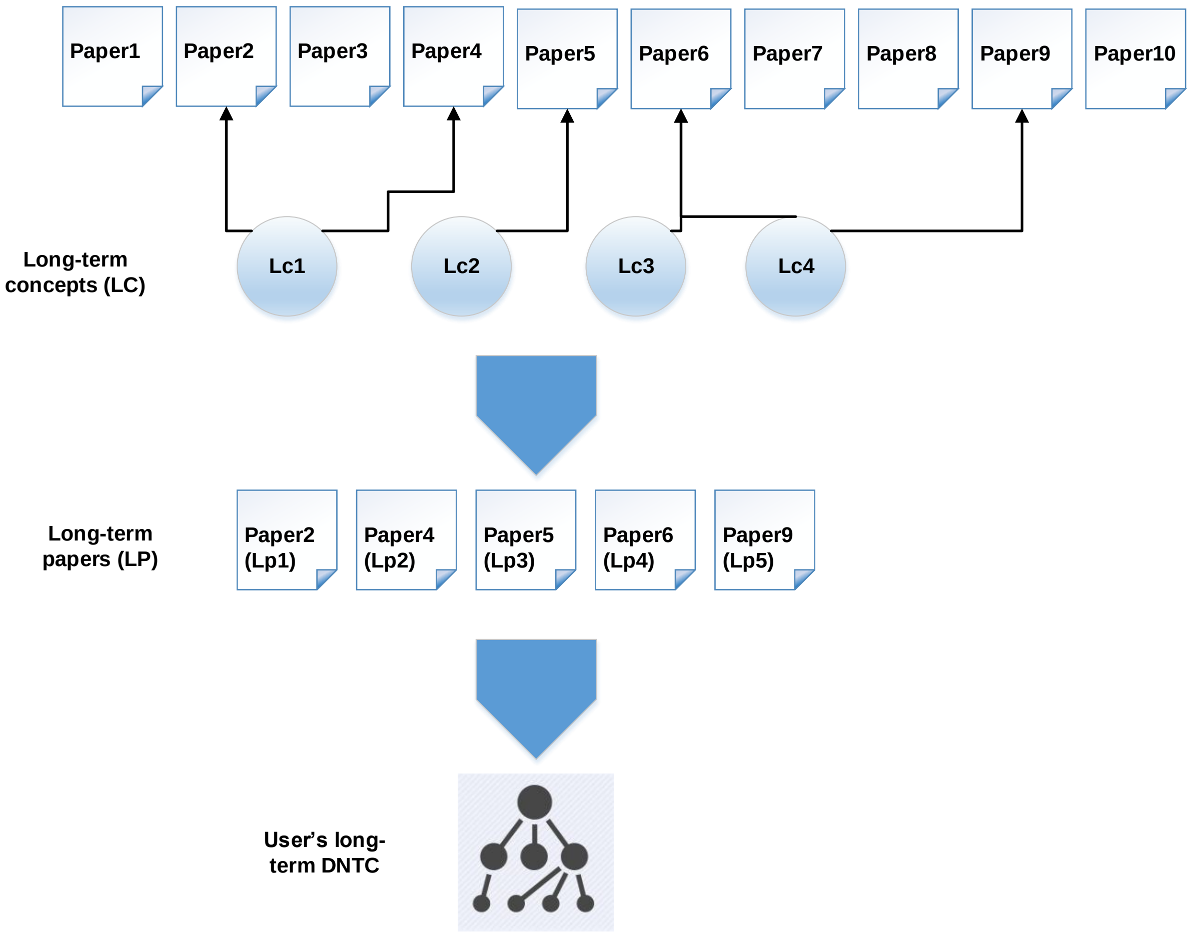

Figure 5.7. An example of the long-term model.

5.3 Evaluations and results

5.3.1 Evaluation of short-term model

We evaluated the performance of our short-term model using the BibSonomy dataset. The BibSonomy dataset in section 5.1 was pruned to remove users with fewer than 60 active days (an active day is a day that the user reads at least one paper). The remaining dataset consists of 1,074 users. Every day in the 60 active days is evaluated for each user. The **training set for an active day *i*** is the research papers in the user's record for previous active days before the active day *i* (i.e. we started the evaluations with active day 2 as shown in appendix B because active day 1 is the first training day), and the **testing set for an active day *i*** is the research papers that exist in day *i* and the next 29 calendar days in the user's records (we assume that the duration for short-term interests is 30 calendar days). At every active day *i*, if a recommended paper exists in its testing set, then it is relevant to his/her short-term interests. The measurement that is used for evaluation is precision at top *k* research papers of an active day *i* for a user *a* as follows:

$$P_k(d_i, a) = \frac{NP_{i,a}}{k} \qquad (5.4)$$

Where $NP_{i,a}$ is the number of recommended research papers that match the testing set for active day *i* for the user *a*. Then, the average precision is calculated for all users (*U*) for an active day *i* as follows:

$$AVG\ P_i = \frac{\sum_{j=1}^{U} P_k(d_i, j)}{U} \qquad (5.5)$$

The mean average precision for all active days is calculated for all active days (*AD*) as follows:

$$MAP = \frac{\sum_{i=1}^{AD} \mathrm{AVG\ P_i}}{AD} \qquad (5.6)$$

5.3.1.1 Evaluating β parameter

In this section we evaluated different values of β (the decay factor in equation 5.2) parameter to find the optimal value that provides the best overall performance for our short-term model. The measurement that is used for evaluation is precision at top 10 research papers (k=10). Figure 5.8 presents the MAP for all users using different values of β in the range of [0.1 to 1]. When β = 0.1, the PDSW length is very small to detect the short-term interests. The results increase when the β value increases until β = 0.6, where the MAP is 0.76. Then, the PDSW length becomes very large and may include some of the old short-term interests that do not belong anymore to the user's current short-term interests. The value of β used in our model was therefore β = 0.6.

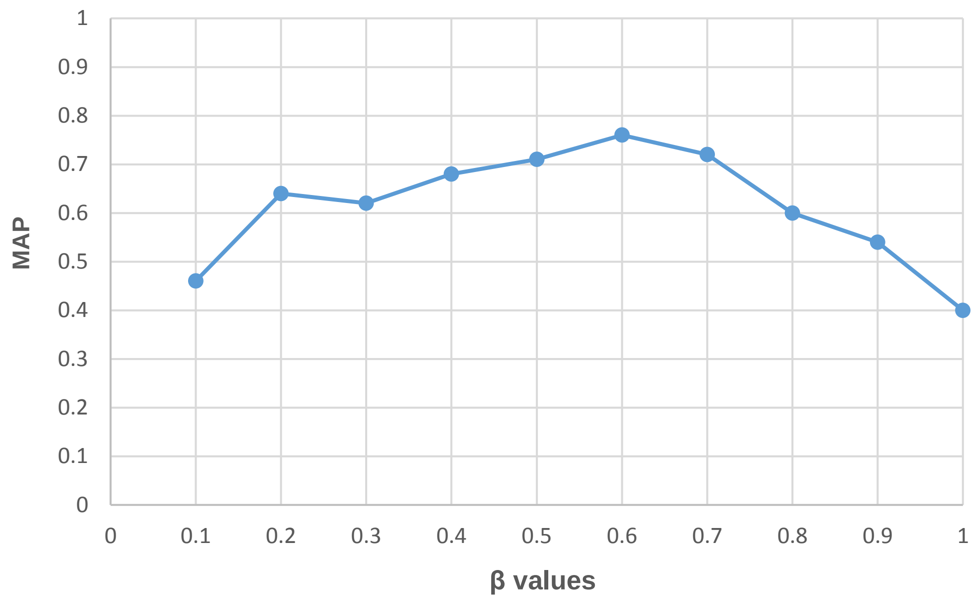

Figure 5.8. MAP results using different β values for the PDSW model.

5.3.1.2 Comparing our short-term model against baselines

We compared our PDSW short-term model against three systems:

1. The DNTC system in Chapter 4.

2. The Static window-time-based model in (Gao et al., 2013).

3. The Dynamic time-based model for the short-term model in (Hawalah and Fasli, 2015).

Our PDSW short-term model and the three systems are run for each day during 60 active days. Figure 5.9 shows the overall comparison for our short-term model against three systems for 60 active days. Table 5.2 shows the MAP that reflects the results of those of Figure 5.9 (appendix B contains the detailed table). It can be seen that the DNTC system achieves the lowest performance with the MAP of 0.47 over the 60 active days. The DNTC system does not consider short-term behaviour but includes all the research papers read by a user. Considering all previous research papers in a user's record give the previous existing concepts high weights in a user's profile, hence they are considered as short-term interests. However, the new concepts receive

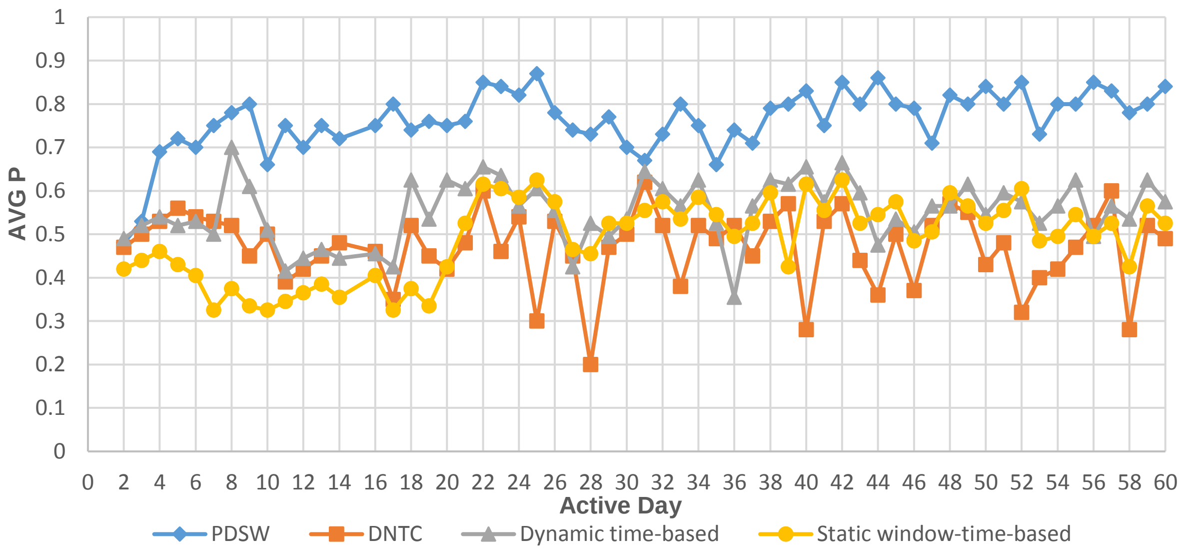

Figure 5.9. Comparing average precision for our PDSW short-term model against three systems.

System	MAP
DNTC	0.47
Static window-time-based	0.49
Dynamic time-based	0.55
PDSW	0.76

Table 5.2. The MAP results for the four short-term systems.

lower weights in a user's profile, which can cause sharp drops in the precision in some active days, for instance, the days 25, 28 and 40. When it comes to the Static window-time-based system, the performance is slightly better than the DNTC system with MAP of 0.49. This is because this system considers only the latest research papers during the static window-time-based. The low performance of this system because it assumes a user's reading behaviour is static, whereas in reality the user behaviour changes over time. Moreover, each user has different personalized behaviour. When it comes to the Dynamic time-based system, there is an improvement in the performance with the MAP of 0.55. This system is better than the previous two systems because it can handle the situation when new short-term concepts arise in a user's profile, and it does not depend on static time-based behaviour. However, it has a limitation that it cannot handle the problem of different inactive days for different users' behaviour. Our PDSW system achieves MAP of 0.75 which is an improvement on each of the previous three systems. These results show that our short-term model can effectively learn different users' reading behaviours even if there are different patterns of inactive days. Moreover, it dynamically adapts to the changes in users' reading behaviour.

5.3.2 Evaluation of long-term model

We evaluated the performance of our long-term model using the BibSonomy dataset. The BibSonomy dataset in section 5.1 was pruned to remove users with fewer than 12 active months during the years 2015 and 2016 (an active month is a month that the user reads at least one paper). The remaining dataset consists of 261 users. Every month in the 12 active months for each user is evaluated. The **training set for**

an active month *i* is the research papers in the user's record for previous active months before the month *i* (i.e. we started the evaluations with active month 2 as shown in table 5.3 because active month 1 is the first training month), and the **testing set for an active month *i*** is the research papers that exist in the rest of the user's record and one of its concepts is long-term concept 'LC'. At every active month *i*, if a recommended paper exists in its testing set, then it is relevant to his/her long-term interests. The measurement that is used for evaluation is precision at top k research papers of an active month *i* for a user a as follows:

$$P_k(m_i, a) = \frac{MPi, a}{k}$$

Where $MP_{i,a}$ is the number of the recommended research papers that exist in the testing set for active month *i* for the user a. Then, average precision is calculated for all users *U* for active month *i* as follows:

$$AVG\ P_i = \frac{\sum_{a=1}^{U} P_k(m_i, a)}{U}$$

The mean average precision for all active months is calculated for all active months (*AM*) as follows:

$$MAP = \frac{\sum_{i=1}^{AM} \mathrm{AVG\ P_i}}{AM}$$

We compared our long-term model against three systems:

1. The DNTC system in chapter 4.
2. The Time-based forgetting factor model in (Gao et al., 2013).
3. The Dynamic time-based for long-term interests in (Hawalah and Fasli, 2015).

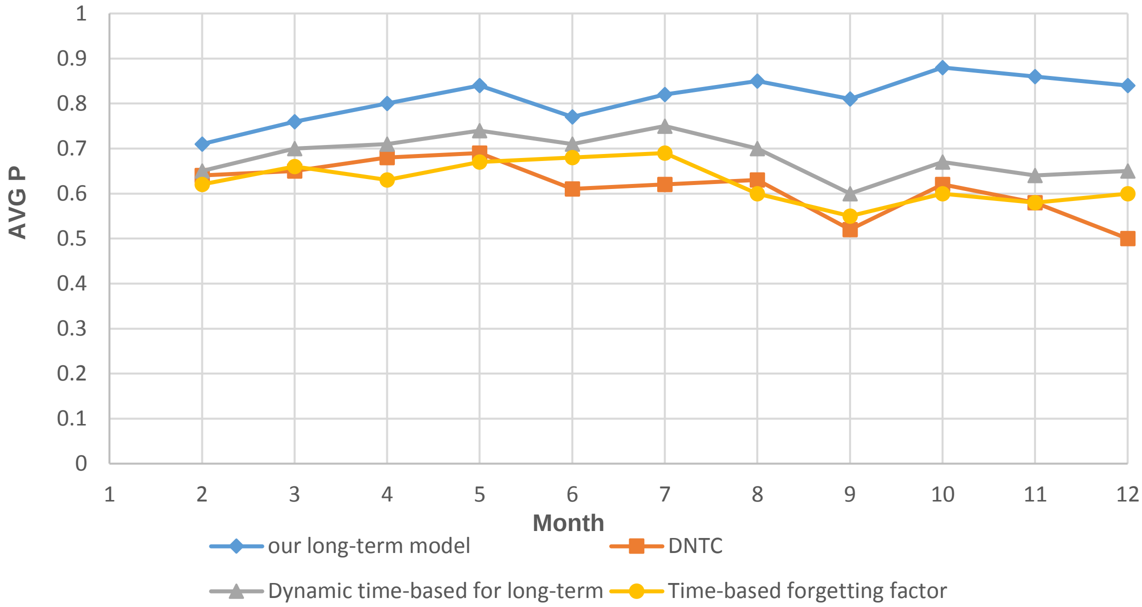

Figure 5.10. Comparing our long-term model against the three systems.

	AVG P			
Active month	**Our long-term model**	**DNTC**	**Dynamic time-based for long-term**	**Time-based forgetting factor**
2	0.71	0.64	0.65	0.62
3	0.76	0.65	0.7	0.66
4	0.8	0.68	0.71	0.63
5	0.84	0.69	0.74	0.67
6	0.77	0.61	0.71	0.68
7	0.82	0.62	0.75	0.69
8	0.85	0.63	0.7	0.6
9	0.81	0.52	0.6	0.55
10	0.88	0.62	0.67	0.6
11	0.86	0.58	0.64	0.58
12	0.84	0.5	0.65	0.6
MAP	**0.81**	**0.61**	**0.68**	**0.63**

Table 5.3. The AVG P and the MAP results for the four long-term systems.

Our long-term model and the three systems are run at the end of each active month for each user. The top 10 recommended research papers (k=10) are evaluated. Figure 5.10 shows the overall comparison for our long-term model against the three systems over 12 months. Table 5.3 shows the AVG P and the MAP that reflect the results of those of Figure 5.10. It can be seen from Figure 5.10 and Table 5.3 that the DNTC system achieves the lower precision performance with the MAP of 0.61. After the fifth month, the performance of the DNTC system declined dramatically because of the cumulative calculations for all the research papers that are read by the user. This low performance is because the DNTC system includes all the research papers in user's records even the research papers for short-term interests. When it comes to the Time-based forgetting factor system, the performance is slightly better than the DNTC system with MAP of 0.63. This is because this system has a forgetting factor. However, this forgetting factor is fixed for all users and does not consider different users' behaviours. When it comes to the Dynamic time-based system for long-term interests, there is an improvement in the performance with the MAP of 0.68. This model is better than the previous two systems because it can handle the situation when there is short-term concepts and long-term concepts, and it does not depend on static time-based technique. However, it has a limitation that it is unable to handle the long inactive periods in users' behaviour. Therefore, after the seventh month, its performance declined significantly. Our long-term model achieves the MAP of 0.81 which is better than each of the previous three systems. This is because our model can effectively learn different users' reading behaviours even if there are different long inactive periods. Moreover, it dynamically adapts to the changes in users' reading behaviour. Our long-term model significantly outperforms the other three baselines after the seventh month as shown in Figure 5.10.

5. 4 Conclusions

In this chapter, we presented our novel short-term and long-term models for a research paper recommender system. First, we analysed users' reading behaviour in the BibSonomy dataset. Our analysis shows that the users' reading of research papers is different to that of reading web pages and news articles. The users have different durations of inactive days and inactive months which can affect the performance of a recommender system that depends on continuous time-based method. Moreover, the number of multiple concepts that are involved in users' reading behaviour has a large distribution of numbers. Some of the concepts are short-term concepts that stay less than one month in a user's record. These concepts do not reflect the long-term interests for a user. Therefore, we developed our short-term and long-term models based on our analysis of users' reading behaviours of the research paper domain. The short-term model is based on the personalized dynamic sliding window (PDSW) that is able to change dynamically according to a user's behaviour changing. The long-term model considers only the long-term concepts and the research papers that belong to these concepts. Our evaluations of the performance show that our models significantly outperform the other baseline systems. Our short-term PDSW model achieves the MAP of 0.76 and our long-term model achieves the MAP of 0.81. The performance advantage is because our models can effectively learn different users' reading behaviours. Moreover, they dynamically adapt to the changes in users' reading behaviour over time. The results from this chapter are published in (Al Alshaikh et al., 2017b). In the next chapter, we will develop a collaborative model, then in chapter 7, we will combine the short-term model, the long-term model and the collaborative model to produce a dynamic hybrid system for the research paper domain.

Chapter 6. Predicting Future Interests in a Research Paper Recommender System Using a Community-Centric Tree of Concepts Model

Most research paper recommender systems suggest research papers which are similar to a user's profile which results in a limited set of recommendations based on current user preferences that are represented in the system (Kotkov et al., 2016). A major challenge in recommender systems is to explore the potential of future interests of users (Yang et al., 2016). Content-based approaches are able to recommend a set of research papers that relate to the user's current interests. However, they suffer from the problem of content overspecialization because they depend only on the metadata of research papers in the user's profile; therefore the user is restricted to getting recommendations similar to the research papers already defined in his/her profile (Isinkaye et al., 2015). Collaborative filtering approaches have the ability to explore potential future interests. Existing collaborative approaches have been developed for domains such as movies, music and e-commerce products. These collaborative approaches are not appropriate for the research paper domain, because they depend on large numbers of user ratings. However, there is a lack of ratings in the research paper domain (Yang et al., 2009). For example, the implicit ratings (users' access logs) on Mendeley (Mendeley, 2014) (research paper domain) has been compared to Netflix (Netflix, 2014) (movie domain), and has been found that the sparsity of Mendeley was three orders of magnitude higher than on Netflix (Beel et al., 2016). This is due to the different behaviour of users in these two domains. For instance, in the movie domain, there are many users who have watched the same movies. Therefore, similar users can be found for most users and hence recommendations can be made effectively. However, the research paper domain suffers from the data sparsity problem, where several new research papers have not been read by any user and further, a new user may read only a few research papers (Jain, 2012 and Beel et al., 2016). This leads to an inability to successfully locate similar users and hence leads to the generation of weak recommendations. In this chapter, we present a new collaborative filtering model that does not depend on users' rating. Our novel method computes the similarity between users according to the users' profiles that are represented as Dynamic

Normalized Trees of Concepts. The similarity between users is computed by using the Tree Edit Distance algorithm. Then, a Community-Centric Tree of concepts (CCT) is created. The CCT is used to recommend a set of research papers that may relate to the user's future interests. The rest of this chapter is organized as follows. Section 6.1 presents our collaborative recommendation model and section 6.2 illustrates evaluations and results using the BibSonomy dataset. Finally, the conclusions are presented in section 6.3.

6.1 Our collaborative recommendation model

The proposed collaborative recommendation model is comprised of three phases:

1- Building the user profiles as Dynamic Normalized Trees of Concepts using the 2012 ACM CCS ontology.
2- Computing the similarity between the target user and candidate users, then generating a "Community-Centric Tree of concepts" (CCT) for the target user.
3- Recommending a ranked list of research papers for the target user based on CCT.

Figure 6.1 presents our collaborative recommendation model and illustrates the three phases. The following subsections explain the phases in details.

6.1.1 Phase 1: Building user profile as DNTC

The main goal of this phase is to build a user profile as Dynamic Normalized Tree of Concepts (DNTC). The BibSonomy dataset is used to create a database of users and the research papers which they have read. This phase involves two steps: classifying the research papers read by the users to the related concepts in the 2012 ACM CCS ontology as in section 4.2 and building a DNTC profile for each user as in section 4.3.

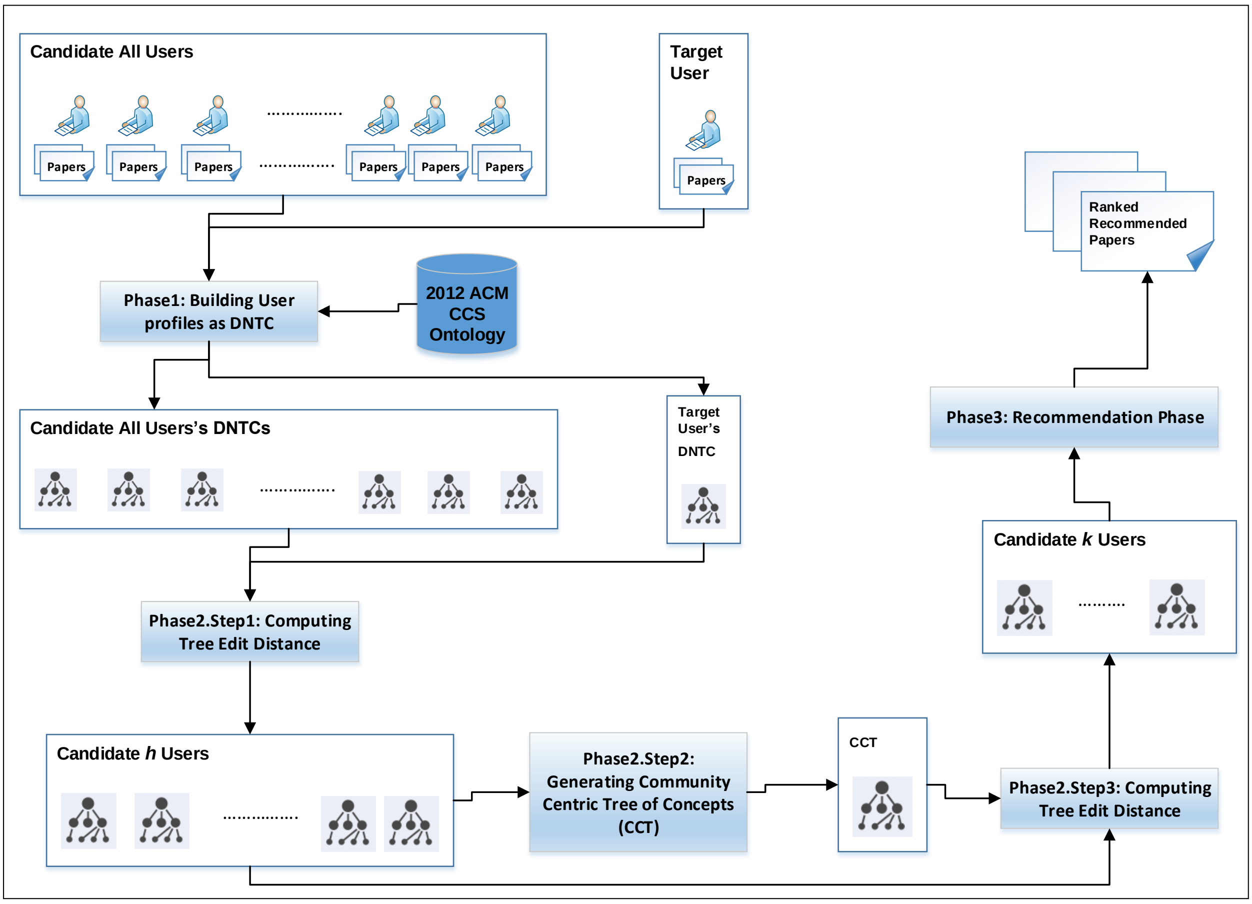

Figure 6.1. Our collaborative recommendation model.

6.1.2 Phase 2: Computing the similarity between users and generating CCT

The purpose of this phase is to determine the community of users whose user profiles are similar to the target user. There are three steps in this phase as follows.

6.1.2.1 Step 1: Find a set of *h* most similar users to a target user

The similarity between a target user and the candidate user is computed using the Tree Edit Distance algorithm to calculate the distance between two DNTC trees, a target user's DNTC and a candidate user's DNTC. The cost of modifying a DNTC tree for a candidate user to match a target user DNTC tree is calculated. The two most similar DNTC trees are those which have the lowest total cost of transformations between them. After calculating the total costs between all DNTC trees for the candidate users and a target user DNTC tree, the total cost together with its associated id of the user (UserID) is stored in a list and the list is sorted in increasing order. Hence, the closest candidate users to the target user appear first in the list and the most distant candidate users appear last. Then, the most h similar users are selected and stored as set h_i for a target user i. h is a parameter that will be evaluated in experiments in section 6.2.2.

6.1.2.2 Step 2: Generating Community-Centric Tree of concepts

The selected h similar users are used to generate a Community-Centric Tree of concepts (CCT). The CCT is generated by combining the h users DNTC profiles as follows. First, $CCTi$ for a target user i is initialized as a tree of the 2012 ACM CCS concepts with zero weights for all concepts. Then, the weights for all concepts from all h similar users are summed up. Finally, all concept weights are divided by the number of h similar users in order to normalize the concept weights. The CCT_i represents the centric of the community interests for the target user i.

6.1.2.3 Step 3: Find the *k* most similar users (from the set *h* users)

In this step, we use $CCTi$ to find the closest users from the set h_i to the centric of the community interests. The similarity between $CCTi$ and the users in the set h_i is computed by using the Tree Edit Distance algorithm. After calculating the total cost between $CCTi$ and the DNTC trees for the users in the set h_i, the total cost with its associated id of the user (UserID) are stored as a list and sorted in increasing order. Hence, the closest user to $CCTi$ appears first and the most distant user appears last. Then, the k most similar users are selected and stored as set k_i for a target user i. The set k_i is a subset of the set h_i. k_i is a parameter that will be evaluated in experiments in section 6.2.2. Evaluation results in section 6.2.2 show that using the set k_i for making recommendations produces better results than using the whole set h_i. This is because the set k_i represents the users that are closer to the CCT_i, which represents the centric of the community interests.

6.1.3 Phase 3: Collaborative recommendation phase

In this phase, a ranked list of the *top N* research papers is recommended to a target user i. First, the research papers that are read by users in the set k_i are retrieved from the database as set Pk_i. If there are any research papers already read by a target user i, then these research papers are removed from the set Pk_i. Then, the set of research papers Pk_i is ranked as follows:

a- If some research papers appear more than once in the set Pk_i, that means there are common research papers between more than one user in the set k_i. The number of appearances of each common paper CP_j in Pk_i is calculated as NCP_j. Then, the research papers in Pk_i are ranked according to NCP_j in descending order. Hence, the most common research papers have higher ranks. We call this ranked list the common research papers list.

b- If there are no common research papers (or the common research papers are fewer than the number of *top N* recommended research papers), then the content-based model is integrated with our collaborative model as follows. We compare the non-common research papers profiles with a target user profile. First, a paper profile is represented as a tree of concepts as in section 4.4. Then, the Tree Edit Distance cost is computed between a target user's DNTC tree and the trees of concepts for the non-common research papers. We order the research papers according to the tree edit distance cost between the paper and the target user's DNTC in increasing order. Hence, the closest research papers to a target user appear first and the most distant research papers appear last. We call this ranked list the non-common research papers list.

The final recommended list that results from the recommendation phase can include both lists: common research papers list and non-common research papers list. The common research papers list appears first before the non-common research papers list. Figure 6.2 shows the flowchart for the collaborative recommendation phase.

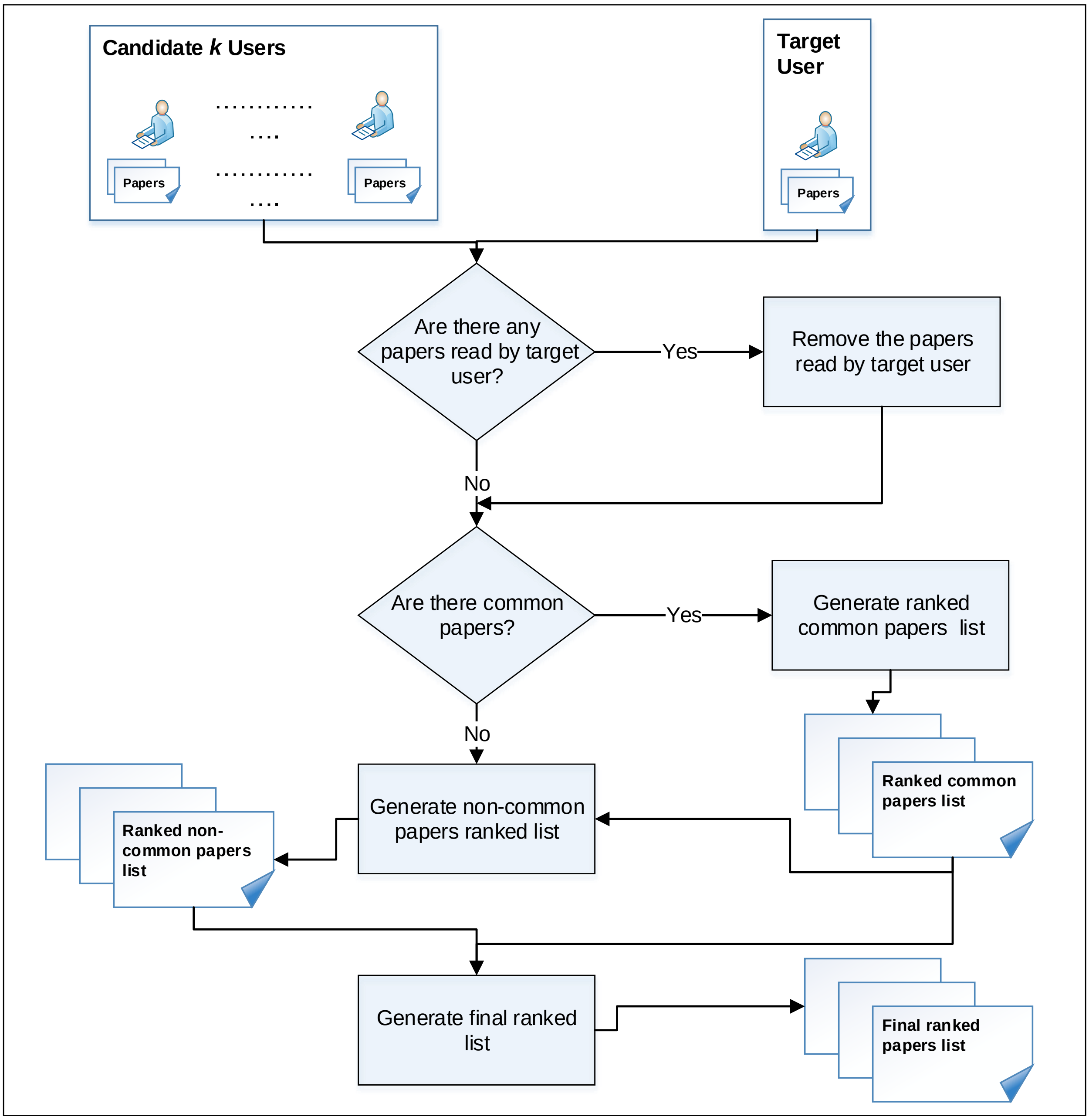

Figure 6.2. Flowchart for the collaborative recommendation phase.

6.2 Evaluations and results

In this section, first the evaluation methodology is explained. Then, our collaborative model parameters are evaluated to find the optimal values. Finally, we compared our proposed collaborative model against two systems.

6.2.1 Evaluation methodology

We evaluated the performance of our proposed collaborative model using the BibSonomy dataset. The users' records for the years 2015 and 2016 for the users in the field of computer and information science are used in the evaluations. This includes 1,642 users and 43,140 research papers. Each paper is classified to the three most closely related concepts from the 2012 ACM CCS ontology. A target user's record is divided into a training set of research papers (60%) and testing set of research papers (40%). The training set is research papers that were read by the user before the testing set. The precision for cut-off results at position N (P_N) is used to evaluate the *top N* recommended research papers. The purpose of this chapter is to evaluate the future concepts for a target user. Therefore, our precision metric for the future concepts of interest is defined as follows.

Assume a set $FC = \{FC_1, FC_2, \ldots\ldots, FC_m\}$ is a set of future concepts, m is the number of future concepts. A future concept is a concept that does not exist in a target user's training set as shown in Figure 6.3. The precision for a future concept (FC_i) is defined as follows:

$$P(FC_i)_N = \frac{Number\ of\ relevant\ recommended\ papers\ to\ FC_i}{N}$$

Then, the average precision (AP_f) for m future concepts for a user is calculated as follows:

$$AP_f = \frac{P(FC_1)_N + P(FC_2)_N + \cdots + P(FC_m)_N}{m}$$

The mean average precision for all users is calculated as follows:

$$MAP_f = \frac{\sum_{i=1}^{U} AP_{fi}}{U}$$

where U is the total number of users. The top 10 recommended research papers are evaluated in our experiments.

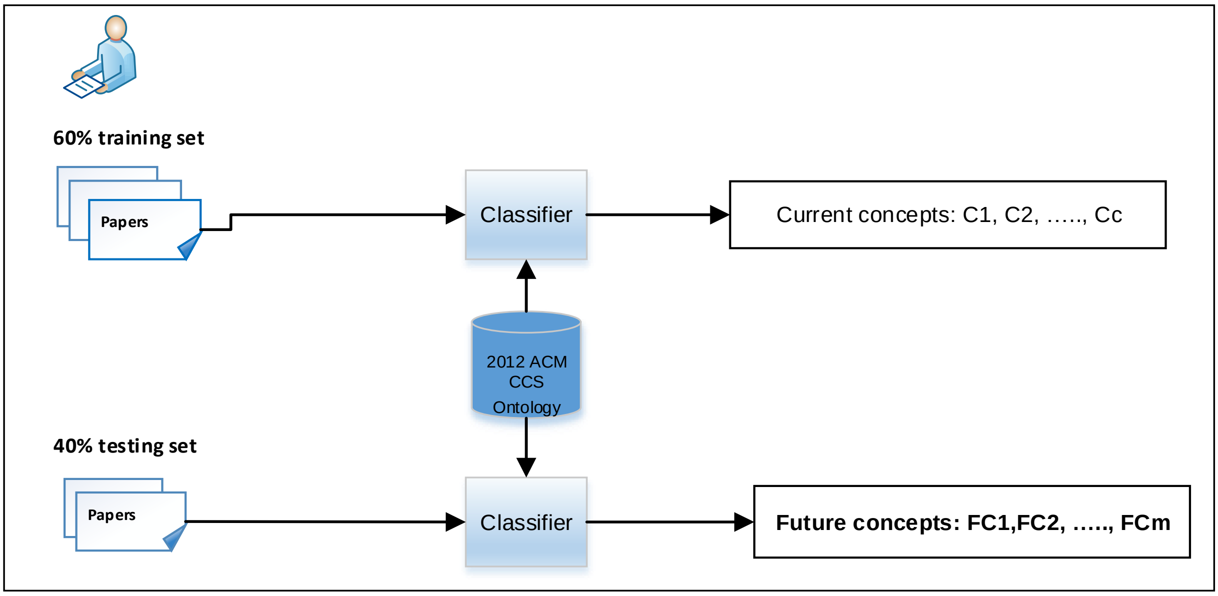

Figure 6.3. Future concepts.

6.2.2 Evaluating our collaborative model parameters

We evaluated our model for two options as follows:

> **Option1:** Without Community-Centric Tree of concepts (Without CCT) (i.e. using the set h of users for recommendation phase).
>
> **Option 2:** With Community-Centric Tree of concepts (With CCT) (i.e. using the set k of users for recommendation phase).

First, we have to find the optimal value for h in option 1, and optimal values for h and k in option 2. Figure 6.4 shows the MAP_f results of applying our recommender system without CCT. Different values for h are tested from 10 to 30 users. It can be clearly seen that the MAP_f results for h = 10 are relatively low. This shows that using the research papers for 10 similar users to be included during recommendation phase is not enough. The MAP_f results increase whenever the h value increases until h=24. When h=24, we have the best result of MAP_f with a score of 0.41. This shows that 24 similar users may hold the most essential concepts in their research papers that are expected to be related to a target user in the future.

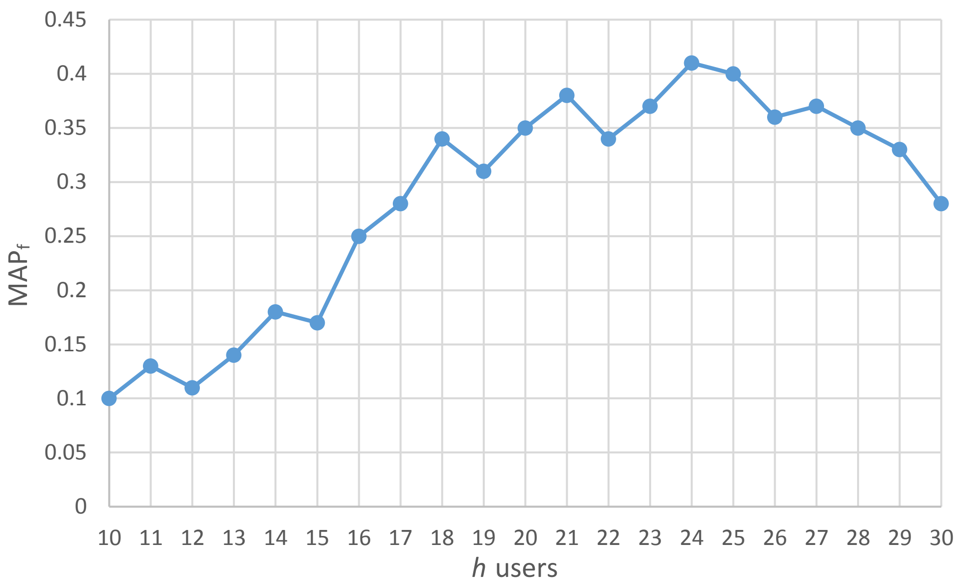

Figure 6.4. *MAP_f* results without CCT for different values of *h*.

Figure 6.5 shows the MAP_f results of applying our recommender system with CCT using different values for *k* and *h*. We tested our system with different values for *h* from 15 to 30 users. It can be clearly seen that the MAP_f results for h = 15 are relatively low. This shows that 15 similar users is a very small number of users to generate CCT using them. The MAP_f results increase whenever the *h* value increases until *h*=21. When *h*=21, we have the best results because 21 similar users may hold the most essential concepts in their research papers to generate CCT. When the *h* value larger than 21, the MAP_f results tend to decrease, this shows that more than 21 similar users is a very large number of users to be included when generating the CCT. We tested our system with different values for *k* from 5 to 12 users. The MAP_f results improve when the *h* value comes close to 21 and *k* values increase. The results are very low when k = 5, this shows that using the research papers for only five similar users during recommendation phase is not enough. In general, the best MAP_f results are when *k*=8, *k*=9 and *k*=10. The optimal MAP_f result is 0.53, when *h*=21 and *k*=9. The results show that the best MAP_f value in option 2 with CCT (MAP_f = 0.53) is greater than the best MAP_f value in option 1 without CCT (MAP_f = 0.41). Therefore, using the CCT provides better recommendations in our system.

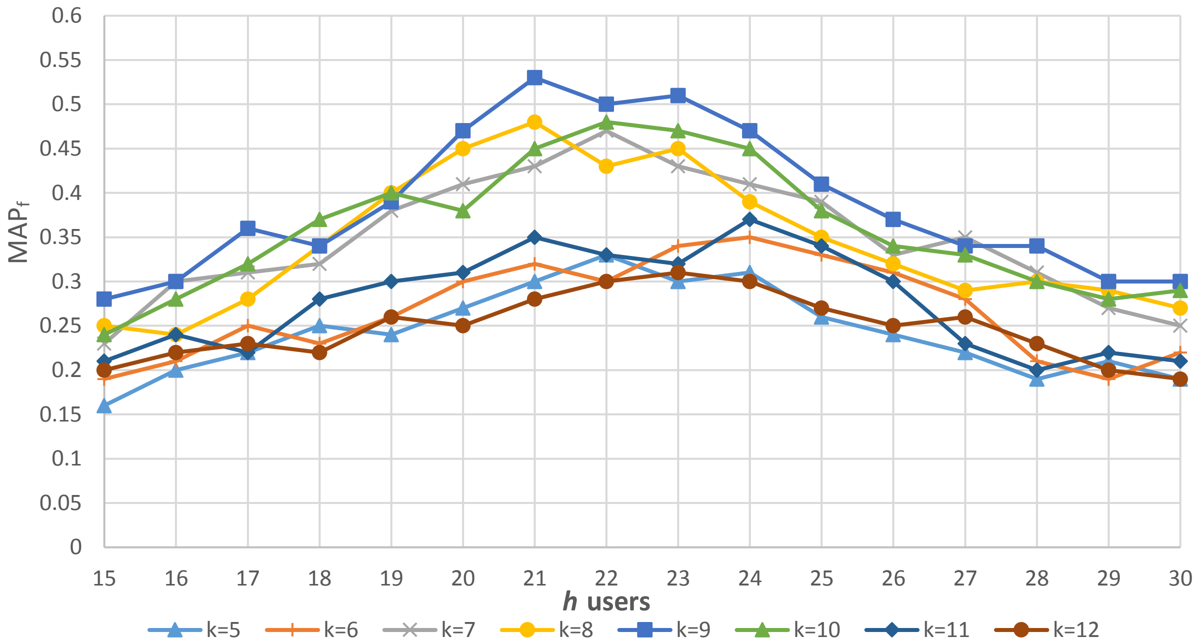

Figure 6.5. *MAP_f* results with CCT for different values of *h* and *k*.

6.2.3 Evaluating our collaborative models against two systems

We compared our proposed model against two systems:

System 1: The content-based DNTC system as in chapter 4: a content-based recommender system that compares a user's DNTC profile with unread research papers' profiles (which are represented as trees of concepts) to recommend the most relevant research papers to the target user's interests. The similarity between a target user and a paper is calculated by Tree Edit Distance algorithm.

System 2: User-based Collaborative Filtering (UBCF) system as in (Nadee et al., 2013): The user-based collaborative filtering model is based on user-item relationships. The similarity between two users is calculated based on the overlap of their paper sets by using the vector cosine similarity algorithm. The

s most similar users are selected. Then, the missing rating for any paper i in target user a is predicted by rating the average from the set of s users' ratings for paper i. The top N research papers that have the highest average rating from the set s similar users are selected to recommend to the target user a. To avoid the problem of the lack of user ratings in the BibSonomy dataset, we assume that if user a did not read paper i, then the rating $r_{a,i} = 0$. If user a read paper i, then the rating $r_{a,i} = 1$. We tested different values of s from 10 to 30 users to find the optimal value of s. Figure 6.6 shows the results for UBCF with different values of s. The best MAP_f is 0.29, when $s = 26$.

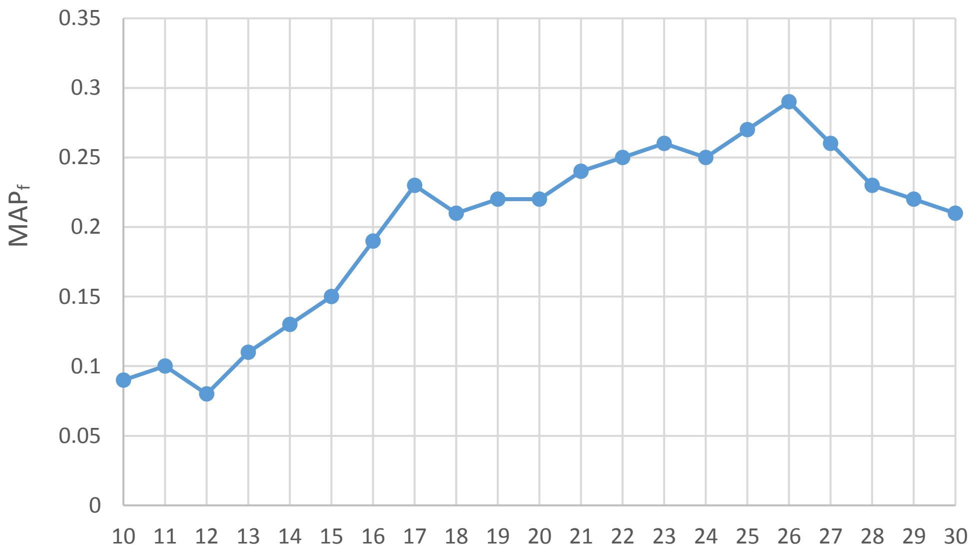

Figure 6.6. Different values of *s* for UBCF model.

Figure 6.7 shows overall comparison results for our collaborative system (with and without CCT) against the other two systems. It can be seen that the DNTC system achieves the lowest precision performance with MAP_f of 0.25. The DNTC system can predict some of the user's future concepts because it maintains the parent-child relationships between the concepts from the 2012 ACM CCS ontology whilst computing the similarity between a user profile and the new research papers to be recommended. However, the DNTC system uses only the current user interests without considering other potential interests that can be extracted from similar users to the target user.

When it comes to the UBCF system, there is an improvement in the performance with the MAP_f to 0.29. This system is better than the DNTC system because it considers potential interests that can be concluded from similar users to the target user. However, it has a limitation of sparsity, because the UBCF system depends on users' rating and the overlap of their paper sets.

Our collaborative system (with and without CCT) outperforms the DNTC system and the UBCF system. This is because it maintains the parent-child relationships between the concepts from the 2012 ACM CCS ontology; considers other potential interests that can be extracted from similar users to the target user; and avoids the problem of sparsity. Our collaborative model with CCT has a better result (i.e. MAP_f =0.53) than our collaborative system without CCT (i.e. MAP_f = 0.41). This is because CCT represents the centric of the community interests.

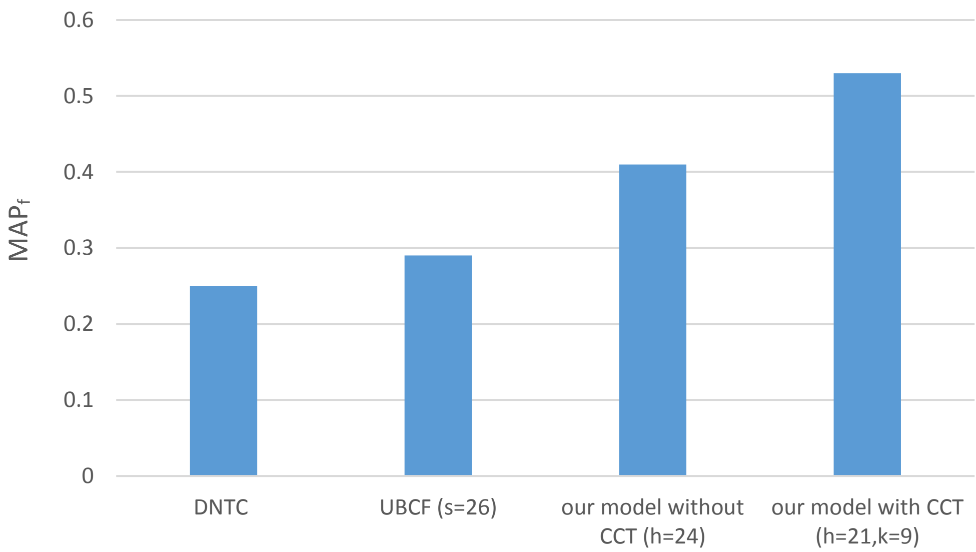

Figure 6.7. MAP_f results for our collaborative system (with and without CCT) against the DNTC and the UBCF systems.

6. 3 Conclusions

The current content-based recommender systems suffer from the problem of overspecialization and they may not have the ability to explore potential future interests. Collaborative filtering approaches can solve this problem; however the existing approaches may not be able to determine similar users which will result in weak recommendations because of the high sparsity problem in the research paper domain. In this chapter, we developed a novel collaborative recommendation method that does not depend on users' rating. Our novel collaborative method computes the similarity between users according to the users' profiles that are represented as Dynamic Normalized Tree of Concepts using the 2012 ACM CCS ontology. Then, a Community-Centric Tree of concepts (CCT) is generated and used to recommend a set of research papers. We performed offline evaluations using the BibSonomy dataset. Different values for the parameters in our collaborative model are tested to find the optimal values. Then our model is compared with two systems: the content-based DNTC and the User-based Collaborative Filtering (UBCF). Our collaborative model (with and without CCT) significantly outperforms the DNTC system and the UBCF system. Our collaborative model with CCT has a better result than our model without CCT. The results from this chapter are published in (Al Alshaikh et al., 2017c). In next chapter, we will integrate our collaborative model with the content-based models that are able to detect short-term and long-term user interests to generate a dynamic hybrid recommender system for the research paper domain.

Chapter 7. A Dynamic Hybrid Research Paper Recommender System

In this thesis we presented our novel techniques to overcome problems or limitations of recommender systems for the research paper domain. In Chapter 4, we developed the DNTC content-based model that is able to determine, maintain and exploit user interests when a user reads a large quantity of research papers and has a large distribution of multiple concepts. In Chapter 5, we proposed the short-term and the long-term content-based models that are able to represent user interests dynamically and adapt to the changes in a user behaviour during his/her short-term and long-term goals. Chapter 6 presented our novel collaborative filtering model to predict user's future interests. A research paper recommender system needs to provide users not only with recommendations for relevant research papers, but also provide these recommendations at the appropriate times whilst the user is researching for information. Therefore, developing a dynamic hybrid system that is able to overcome the problems and limitations in the research paper domain is important. Nevertheless, integrating all our previous models to provide an effective dynamic personalization system is a complex task. This is because we have to find the right balance and cooperation between all our previous models. In the hybrid system, we endeavour to answer these questions: (1) how to represent multiple user interests; (2) how to merge multiple recommendation lists to be one unified list; and (3) how to rank the unified recommendation list. In this chapter, our objective is to develop a dynamic hybrid research paper recommender system that can integrate and exploit the content-based models for short-term and long-term interests with the collaborative model to provide a user with a recommendation list that contains the most related research papers to his/her interests at the appropriate time. In section 7.1 our dynamic hybrid research paper (DHRP) recommender system is presented. Then, the evaluations and results for the DHRP system are illustrated in section 7.2. Moreover, in this chapter we innovate a new ranking measure to evaluate the ranking performance of a recommender system for multiple concepts in section 7.3. Then, the conclusions for this chapter are discussed in section 7.4.

7.1 Dynamic hybrid research paper (DHRP) recommender system

Content-based systems deal with just current user interests, while collaborative systems deal with future user interests. The former systems assume that users would have the same interests as in his/her profile, whereas the latter systems usually focus on the future interests that do not exist in the user current profile in order to determine future interests from similar users to a target user. We argue that all types of user interests (current short-term interests, current long-term interests and future interests) should be taken into account when providing personalized research papers recommendations. This is because in real life, a user may have different types of interests and preferences, and hence effective recommender systems should be able to model different types of user interests to recommend the right research papers at the right time. In this section, we propose our novel DHRP system that maintains and integrates different types of user dynamic interests. This system was motivated by the following requirements:

- The DHRP system should be able to capture, model and exploit multiple types of the user's interests.
- The DHRP system should be able to adapt its recommendation to any changes in a user's behaviour.

To address the challenges and requirements in the research paper domain, in this chapter we integrate all the previously proposed methods and techniques in chapters 4, 5 and 6 to provide dynamic personalization research papers recommendation system that adapts to different users' behaviours and interests. Figure 7.1 illustrates the main architecture of our DHRP system. In this figure, the three recommendation lists from our previous models are used:

1- The PDSW recommendation list, which resulted from our content-based short-term interests model in chapter 5.
2- The long-term recommendation list, which resulted from our content-based long-term interests model in chapter 5.
3- CCT recommendation list, which resulted from our collaborative model with the community-centric tree of concepts in chapter 6.

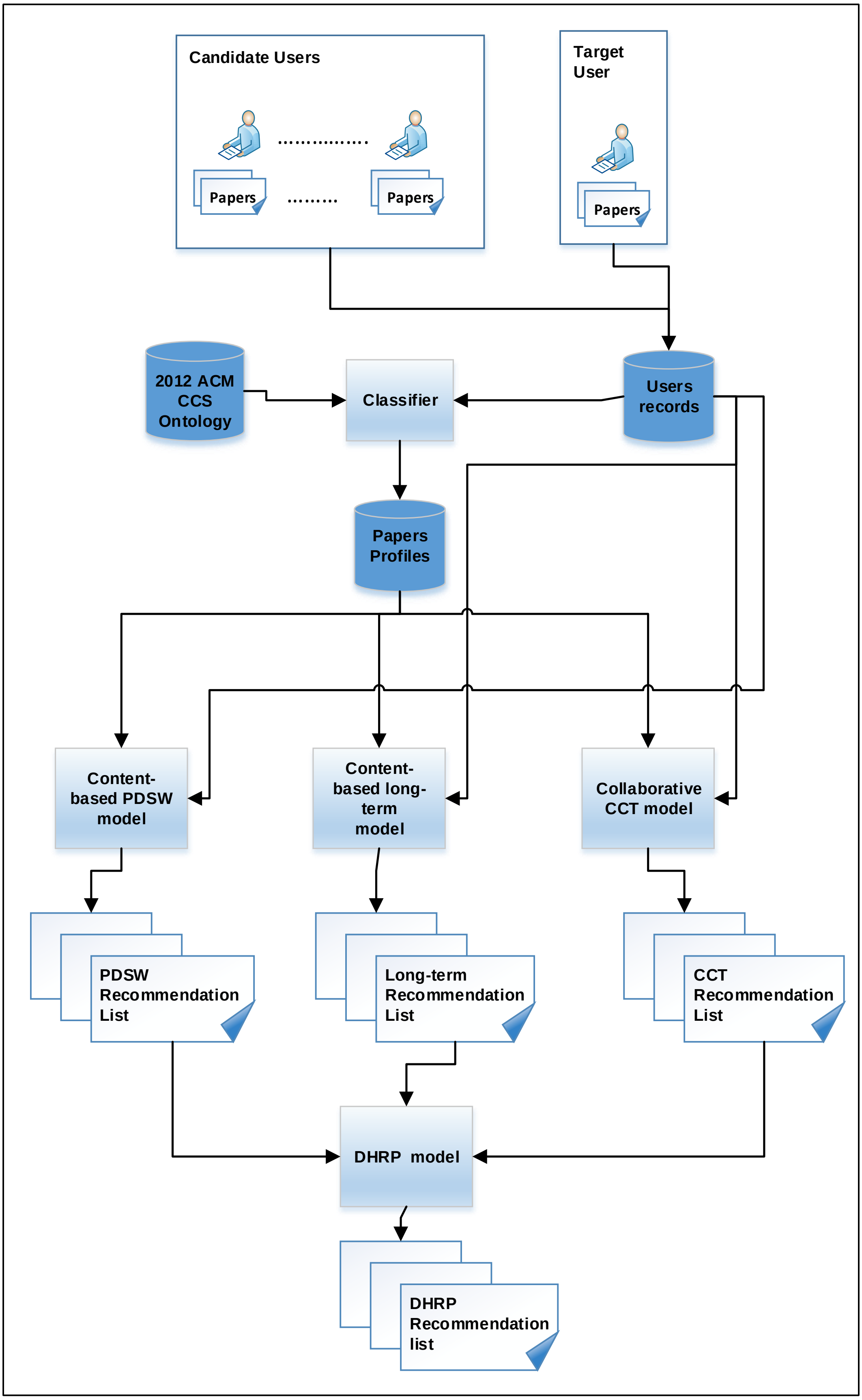

Figure 7.1. The main architecture of the DHRP system.

All these three models are built based on the user's profile based on DNTC user profiling model presented in chapter 4. However, each individual model focuses on specific types of interests. The three recommendation lists might have the same results suggested by the different models. In Figure 7.2, for example, we can see that there are some common results. In this figure, there are 7 different sections created from the intersections between the recommended lists. These sections represent the following:

- Section 1: common research papers between all the three recommendation lists.
- Section 2: common research papers between the collaborative CCT model and the content-based PDSW short-term model.
- Section 3: common research papers between the collaborative CCT model and the content-based long-term model.
- Section 4: common research papers between the content-based long-term model and the content-based PDSW short-term model.
- Section 5: research papers that belong only to the content-based PDSW short-term model.
- Section 6: research papers that belong only to the content-based long-term model.
- Section 7: research papers that belong only to the collaborative CCT model.

These sections do not always appear, for example, in some cases, there are no intersections between some or all of the recommendation lists. In order to provide a user with the right research papers at the appropriate time, we need a mechanism to integrate these lists and just select the research papers which are highly relevant to the user at that time.

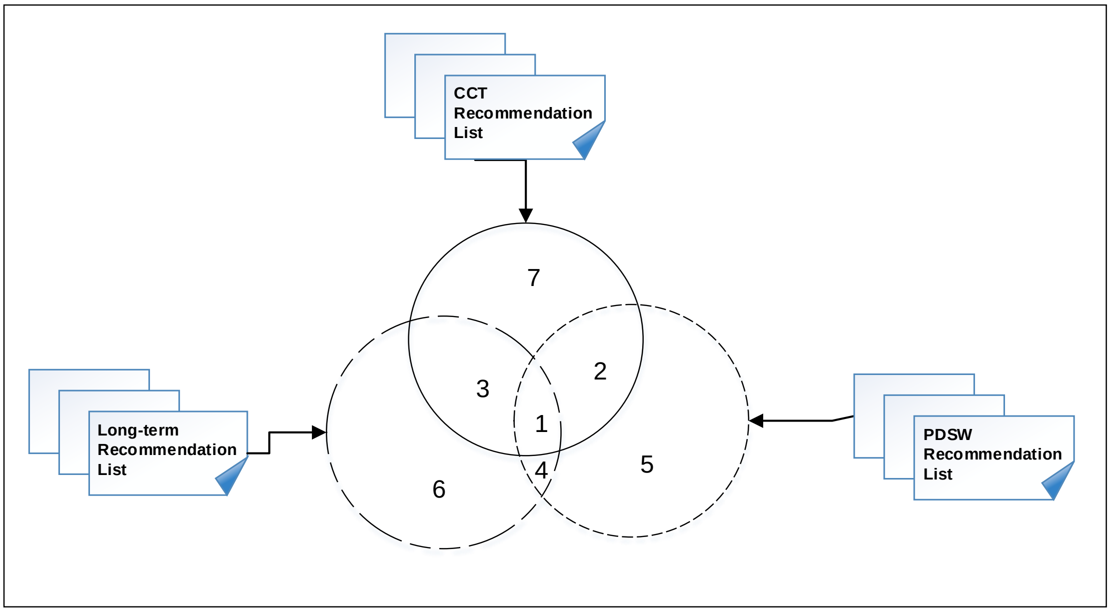

Figure 7.2. Example of common results between the three recommendation lists.

In our DHRP hybrid model, we present integrating all the results from all the recommendation lists and merge them into one unified list, then rank the results. If one paper occurs in more than one list, the weight of such paper would be the mean of all its weights in all lists. Formally, this mechanism can be seen as the mean of all recommendation lists. This mechanism has advantageous in case one of the models has no well information on specific time (e.g. a user has no clear long-term interests, or has no clear future concepts) as this mechanism can deal with this situation by suggesting the research papers that are highly relevant to the available information on that time. Two types of the mean are tested in our DHRP system: arithmetic mean and harmonic mean. Arithmetic mean and harmonic mean can be given as follows:

$$A_p = \frac{1}{n}\sum_{i=1}^{n} s_i \qquad (7.1)$$

$$H_p = \frac{n}{\sum_{i=1}^{n}\frac{1}{s_i}} \qquad (7.2)$$

Where A_p is the arithmetic mean for a paper p and H_p is the harmonic mean for the paper p. s_i is the similarity cost for the paper p and n is the number of appearance of the paper p in the three lists $(1 \leq n \leq 3)$. The similarity cost s_i for a paper p is the tree edit distance cost that is resulted in the content-based model (short-term model or long-term model) along with a paper p and ranked according to it. However, in the collaborative model list, each paper is ranked according to the number of appearances (NCP_p) in the candidate k similar users. Therefore, the similarity cost s_i for a paper p in the collaborative list is calculated as follows:

$$s_i = \frac{Edit\ distance\ cost\ between\ the\ paper\ p\ tree\ and\ the\ user\ DNTC\ tree}{NCPp + \gamma} \qquad (7.3)$$

Where ϒ is the importance factor for the collaborative model results. After calculating the similarity cost for all research papers in all lists, the harmonic mean (or arithmetic mean) is calculated. Finally, all the research papers from all the lists are merged in one unified list and ranked according to their harmonic mean (or arithmetic mean). Then, the DHRP system recommends the top k research papers with the highest harmonic mean (or arithmetic mean) values to a target user. The evaluation results in section 7.2 show that our hybrid system that ranks the results according to the harmonic mean provides better recommendation than our hybrid system with the arithmetic mean.

7.2 Evaluations and results for the dynamic hybrid system

First, the evaluation methodology is explained. Then, the DHRP model's parameter (ϒ) and two types of means (arithmetic mean and harmonic mean) are evaluated to find the optimal value for ϒ and the best results. Finally, we compared our proposed hybrid system against our individual models.

7.2.1 Evaluation methodology

We evaluated the performance of the proposed DHRP hybrid model using the BibSonomy and the CiteSeerX datasets. The users' records in the BibSonomy dataset are used to validate the recommendations. The users who have less than 60 active days are removed from the users set. The remaining users set consists of 1,074 users. For the research papers recommendation, we used 43,140 research papers from the BibSonomy dataset and 100,000 research papers from the CiteSeerX dataset. 2,170 redundant research papers are removed by comparing the titles of the research papers. Therefore, the total research papers set contains 140,970 research papers.

Our hybrid model is run for each day during 60 active days. The **training set for an active day *i*** is the research papers in the user's record for previous active days before the active day *i* (i.e. we started the evaluations with active day 2 as shown in appendix C because active day 1 is the first training day), and the **testing set for an active day *i*** is the research papers that exist **only** in the active day *i*. We use dynamic evaluation day by day. At every active day *i*, if a recommended paper is relevant to at least to one concept of the *concepts* that exist in the research papers in the testing set, then it is relevant to the user's interests. Assume a set $C_i = \{c_1, c_2, \ldots\ldots, c_m\}$ is a set of the concepts that exist in the research papers in the testing set for the active day *i*, *m* is the number of the concepts that are interested by the user *a* in the active day *i*.

The measurement that is used for evaluation is precision for the concept c_j at top *k* research papers as follows:

$$P_k(c_j) = \frac{Number\ of\ relevant\ recommended\ papers\ to\ concept\ c_j}{k} \quad (7.4)$$

Where c_j belong to the interesting concepts C_i in an active day *i*. Then, the average precision for the *m* concepts for the user *a* at the active day *i* is calculated as follows:

$$AvgP_k(d_i, a) = \frac{P_k(c_1)+P_k(c_2)+\cdots+P_k(c_m)}{m} \qquad (7.5)$$

Then, the mean average precision (*MAP*) for all users (*U*) for active day *i* is calculated as follows:

$$MAP_i = \frac{\sum_{j=1}^{U} AvgP_k(d_i,j)}{U} \qquad (7.6)$$

Finally, for a single value for a system we calculate the average of MAP_i for all active days (*AD*) for all users as follows:

$$AVG\_MAP = \frac{\sum_{i=1}^{AD} MAP_i}{AD}$$

We evaluated the top 10 recommended research papers (*k*=10).

7.2.2 Evaluating HDRP model parameters

We evaluated our DHRP model using two types of means: arithmetic mean and harmonic mean, with different values of ϒ parameter (the importance factor for the collaborative list in equation 7.3) to find the optimal value that provides the best overall performance for our hybrid recommender system.

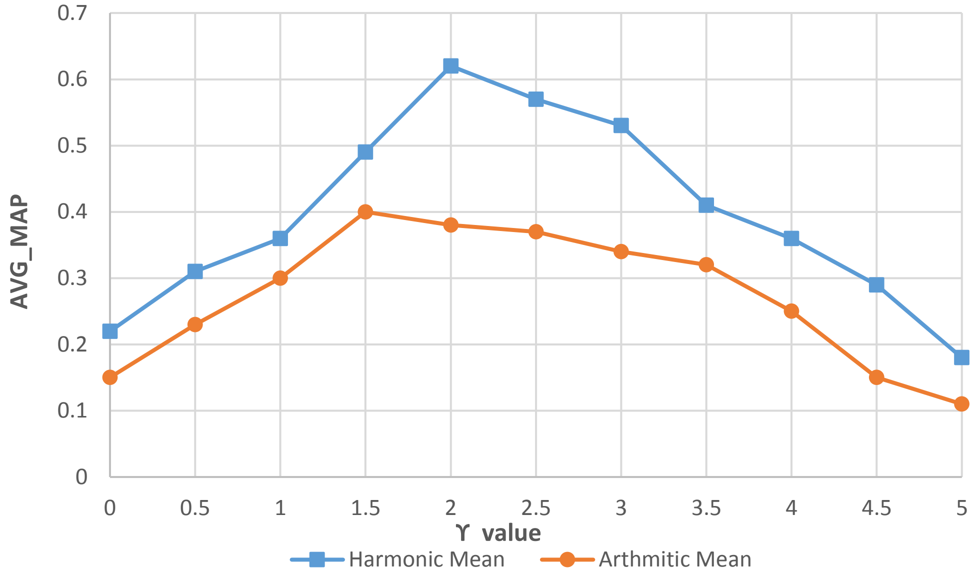

Figure 7.3. Our DHRP model using the harmonic mean and the arithmetic mean with different values of ϒ.

Figure 7.3 shows the AVG_MAP for all users for all active days. We can see that the results for the harmonic mean are better than the arithmetic mean. The harmonic mean is better because for some situations there are extreme outliers exist in the similarity costs. If there is a similarity cost that is much higher than the rest (outlier), the harmonic mean is the appropriate average to use. Unlike the arithmetic mean, the harmonic mean gives less significance to the high-value outliers to provide a truer picture of the average.

We tested our DHRP model with different values for ϒ in the range of [0 to 5]. Figure 7.3 shows that the AVG_MAP results improve when ϒ value between 1.5

and 2.5. The results clearly tend to decrease when ϒ reach the smallest or largest values (i.e. 0 and 5 respectively). The worst results are when ϒ=5, this is because dividing the tree edit distance cost for a paper by a large number produces a very low similarity cost (s_i) in equation (7.3), which leads to rising the collaborative list to be more important than the content-based lists in the final ranked hybrid list. The hybrid model should balance the importance of the content-based lists and the collaborative list to provide the best results that meet the users' interests. The best AVG_MAP result for harmonic mean is when ϒ= 2 with AVG_MAP =0.62. The best AVG_MAP result for the arithmetic mean is when ϒ= 1.5 with AVG_MAP =0.4. For the next experiment, our hybrid recommender system is used with the harmonic mean and ϒ= 2.

7.2.3 Evaluating our hybrid model against our individual models

Our hybrid model is compared against our individual models not against baselines because to the best of our knowledge, there is no hybrid system as baseline for the research paper domain and we have shown previously in chapters 5 and 6 that the recommender systems for other domains such as web pages or movies are not applicable to the research paper domain. Moreover, we want to validate the argument of using the hybrid system is providing better recommendations than using only the content-based model or the collaborative model. We compared our hybrid system against our three individual systems: the content-based short-term (CBS) system, the content-based long-term (CBL) system and the collaborative system. Figure 7.4 shows the overall comparison for our hybrid model against our individual models over 60 active days.

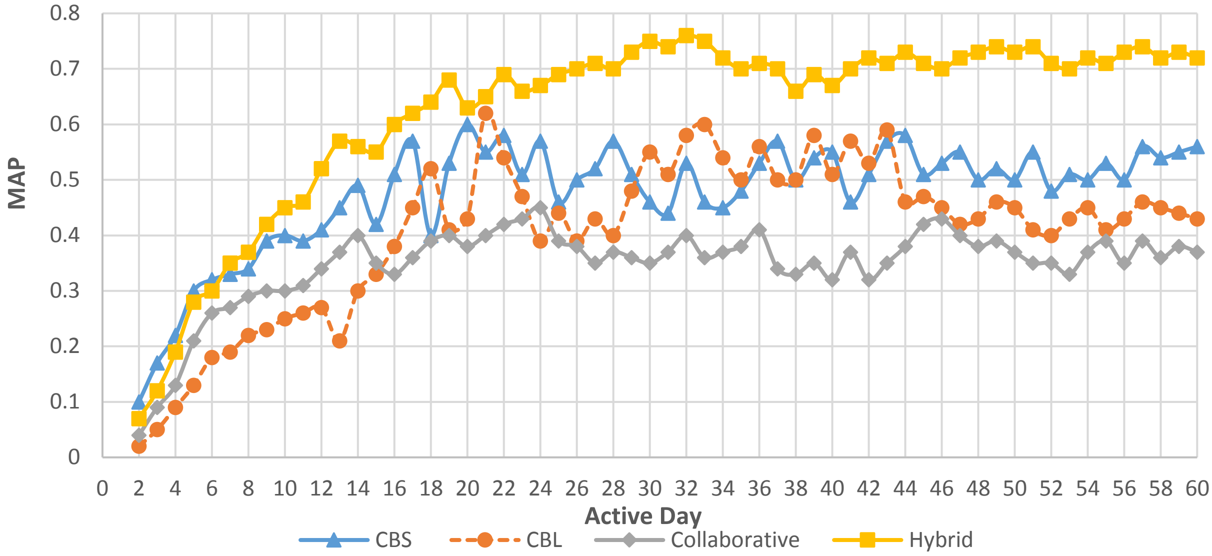

Figure 7.4. Comparing the MAP for our hybrid model against our individual models. (See appendix C for the detailed table.)

The results in Figure 7.4 show that at initialization the system (start-up period where there is no previous knowledge about the user's interests) until the sixth day, the CBS model has the best performance. This is because the CBS model is able to quickly detect the short-term concepts, whereas the hybrid model is affected by the long-term model and the collaborative model that require more information and training during the system start-up period. After the sixth day, it can be seen that all the individual models improved, and the hybrid model improved according to the individual models' improvements. The hybrid model has the best performance against the individual models. After 23 active days, the hybrid system is steady and stable compared with the individual models. The individual models unstable and have dips in their performance at some days. For example, the CBS model has dips in performance at the days 25, 31, 34, 41 and 52. The CBL model has dips in performance at the days 24, 35, 44 and 52. The collaborative model has dips in performance at the days 27, 38, 40, 42 and 53. These dips in the performance of the individual models may be due to the shift or drift of concepts of users' interests. Our hybrid system is able to adapt to users' shift or drift of interests.

Table 7.1 illustrates the AVG_MAP over three types of periods: overall 60 active days; without system start-up period; and during the stable period. In the second column in Table 7.1, it can be seen that the hybrid model achieves the highest AVG_MAP with 0.62 over 60 active days. The individual models have lower AVG_MAP results than the hybrid model because each one of them focuses on a specific type of interests and unable to adapt rapidly to the shift or drift of some interests. All these types of interests and multiple concepts are cooperative together in the hybrid model to provide the best recommendation list.

System	**AVG_MAP over all 60 days**	**AVG_MAP without system start-up period (after 6 days)**	**AVG_MAP during the stable period (after 23 days)**
Hybrid model	0.62	0.66	0.72
CBS model	0.48	0.5	0.51
CBL model	0.41	0.43	0.48
Collaborative model	0.35	0.36	0.37

Table 7.1. Comparing all our models during three types of periods.

The third column in Table 7.1 shows the AVG_MAP for all models without the system start-up period (after 6 days). The results for all the models are better without considering the start-up period because during this period the systems have no previous knowledge about the users and they start building up the users' profiles. The best performance is for the hybrid model with AVG_MAP of 0.66. The last column in Table 7.1 presents the AVG_MAP for all models during the stable period. We consider the stable period is after 23 days. It can be seen that during the stable period the hybrid system has a clear improvement in the results with AVG_MAP of 0.72. For deeper analysis, we can see that both content-based models (CBS and CBL) have better performance than the collaborative model. For example, in the last column in Table 7.1, the CBS model has AVG_MAP with 0.51 and the CBL model with 0.48, whereas the collaborative model has the lowest result with 0.37. This may be because the concepts that are resulted from a user's own profile from the content-based model are more important than the concepts that are from other users' profiles as a result of the collaborative model. Nonetheless, all these concepts provide a better understanding of the user's needs in the hybrid system to produce a recommendation list which best meets the user's interests.

7.3 New ranking measure for multiple concepts

One of the main advantages of the dynamic hybrid model is the ranking of the results according to the harmonic mean to provide better recommendations to users. Consequently, we have the need to measure the improvement in the ranking performance for the dynamic hybrid system against the other systems. The Normalized Discounted Cumulative Gain (NDCG) is one of the measures widely used in information retrieval to evaluate the performance of recommender systems (Shani and Gunawardana, 2011, Jannach et al., 2010 and Beel et al., 2016). However, the NDCG metric is not designed to measure the performance for multiple concepts. In this section, we will discuss the limitation of the NDCG and how we can solve this limitation. Section 7.3.1 presents the properties of the NDCG metric in a recommender system. Then, in section 7.3.2 the limitation is discussed with an example. After that, our solution and the proposed ranking measure is discussed in section 7.3.3. Afterward, the evaluation results using the new proposed ranking measure for the dynamic hybrid system against the individual systems are illustrated in section 7.3.4.

7.3.1 NDCG properties in a recommender system

The main advantage of the NDCG metric is that it allows a discount function over the ranking of the recommendation list (Wang et al., 2013). This feature is very important for recommender systems because the highest ranked research papers are more important than the others. The NDCG is a normalization of the Discounted Cumulative Gain (DCG) measure as follows:

$$NDCG = \frac{DCG}{IDCG} \qquad (7.7)$$

Recommender systems can use a cut-off top-k version of the NDCG. Such NDCG measure is referred to as $NDCG_k$. The DCG is a weighted sum of the degree of relevancy of the ranked papers. The weight is a decreasing function of the rank (position) of the paper, and therefore called discount (Wang et al., 2013). The DCG is the logarithmic discount as follows (McSherry and Najork, 2008):

$$DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i - 1}}{log_2(i+1)} \qquad (7.8)$$

Where k is the number of recommended papers, i is the rank, rel_i is the relevance of the recommended paper. In our evaluation method the relevance values are {0, 1}. The value 0 means irrelevant and 1 is relevant. The normalization of the DCG into the range [0, 1] is done by dividing the DCG values with IDCG, which is the maximum possible (ideal) DCG for a given set of papers and relevancies. The standard IDCG for a set of relevance values {0, 1} is:

$$IDCG = \sum_{i=1}^{n} \frac{2^{rel_i - 1}}{log_2(i+1)} \qquad (7.9)$$

Where n is the total number of only relevant papers (i.e. papers with $rel_i = 1$). Hence,

$$IDCG = \sum_{i=1}^{n} \frac{2^1 - 1}{log_2(i+1)} = \sum_{i=1}^{n} \frac{1}{log_2(i+1)} \qquad (7.10)$$

The NDCG will be used to measure the performance of recommender systems which use multiple concepts for the top k retrieved papers. Let $P_{1,} P_{2,} \ldots\ldots, P_k$ be k ranked recommended papers, let O be a set of concepts from the 2012 ACM CCS ontology (or another ontology). We assume that every paper P_i must belong to exactly one concept c_j ($c_j \epsilon\ O$), which is a concept with a higher weight that is determined by our classifier for the paper P_i. We assume a user has m interesting concepts. Hence, let C be a set of concepts that are interested to the user C= {$c_1, c_2, \ldots., c_m$ } where m is the number of the interested concepts. Hence, the DCG_k for a concept c_j (i.e. ($c_j \epsilon\ C$)) is:

$$DCG(c_j)_k = \sum_{i=1}^{k} \frac{2^{rel_i(c_j)}-1}{log_2(i+1)} \qquad (7.11)$$

Where $rel_i(c_j)$ is the relevance value of a paper P_i to a concept c_j. The standard IDCG for multiple concepts is:

$$IDCG(c_j)_k = \sum_{i=1}^{n\,(c_j)} \frac{1}{log_2(i+1)} \qquad (7.12)$$

Where $n\,(c_j)$ is the total number of relevant papers to a concept c_j.

7.3.2 The limitation of the standard IDCG

There is a limitation in the standard IDCG to measure the performance for a recommender system for multiple concepts. If we used the standard IDCG to normalize the DCG, then the NDCG metric does not measure the properties that we want for multiple concepts. The following example explains this limitation:

Example 1:

Assume that $\{P_1, P_2, \ldots\ldots, P_{10}\}$ are the top 10 retrieved papers (i.e. k=10). Where P_1 is the paper with rank 1. We assume a user has five interested concepts C= $\{c_1, c_2, c_3, c_4, c_5\}$. Assume the following results are from the classifier:

$\{P_1, P_2, P_3, P_4\}$ belong to c_1

$\{P_5, P_6, P_8\}$ belong to c_2

$\{P_7\}$ belong to c_3

$\{P_9\}$ belong to c_4

$\{P_{10}\}$ belong to c_5

Let us focus and measure the first concept c_1. The DCG for the concept c_1 is:

$$DCG(c_1)_{10} = \sum_{i=1}^{10} \frac{2^{rel_i(c_1)}-1}{log_2(i+1)} = \frac{1}{log_2(2)} + \frac{1}{log_2(3)} + \frac{1}{log_2(4)} + \frac{1}{log_2(5)} = 2.56$$

The standard IDCG for the concepts c_1 is:

$$IDCG\ (c_1)_{10} = \sum_{i=1}^{4} \frac{1}{log_2(i+1)} = 2.56$$

Hence, the normalized DCG for the concept c_1 is:

$$NDCG(c_1)_{10} = \frac{DCG(c_1)_{10}}{IDCG\ (c_1)_{10}} = 1$$

This result does not represent a good measure for the multiple concepts algorithm. For example, if we assume that $NDCG(c_1)_{10} = 1$, then the ideal result for c_1 is when the algorithm recommends only the first 4 papers for c_1, whereas we assume that the ideal result for c_1 is when the algorithm recommends all first 10 papers for the concept c_1. Moreover, if the algorithm recommends only the first 2 or 3 papers for the concept c_1 , then the value of $NDCG(c_1)_{10}$ will be the same = 1 (i.e. if the recommendation algorithm returns two results with relevant values {1,1,0,0,0,0,0,0,0,0} and {1,1,1,0,0,0,0,0,0,0} respectively for the concept c_1, both would be considered equally good). This problem occurs because the ideal DCG (IDCG) is not ideal for the evaluation of multiple concepts.

7.3.3 Our solution for the limitation of the standard IDCG

The actual purpose of using the IDCG is to normalize the DCG values in the range [0, 1] to get the NDCG. We suggest that IDCG equation need to be changed to give us normalized results for the DCG to measure multiple concepts. We want these results to be between [0, 1] and the summation of the NDCGs for multiple concepts to be 1. Therefore, we have to find the IDCG equation that provides us with a good measure for multiple concepts. Let the following equation represents the summation of M_NDCGs for multiple concepts to have the value 1 for the top k recommended papers:

$$\sum_{j=1}^{m} M\_NDCG\left(c_j\right)_k = 1 \qquad (7.13)$$

Now we can substitute the NDCG with equation 7.7 as follows:

$$\frac{\sum_{j=1}^{m} DCG\left(c_j\right)_k}{M\_IDCG_k} = 1 \qquad (7.14)$$

Then, the value of M_IDCG$_k$ is calculated for multiple concepts from the above equation 7.14:

$$\mathrm{M\_IDCG}_k = \sum_{j=1}^{m} \mathrm{DCG}(c_j)_k \qquad (7.15)$$

Now we can substitute $\mathrm{DCG}(c_j)_k$ with its equation 7.11 as follows:

$$M\_IDCG_k = \sum_{j=1}^{m} \sum_{i=1}^{k} \frac{2^{rel_i(c_j)}-1}{log_2(i+1)} \qquad (7.16)$$

Where *m* is the number of concepts and *k* is the top *k* papers. Then, we decode the summation for the multiple concepts c_j from *j=1* to *m*:

$$M\_IDCG_k = \sum_{i=1}^{k} \frac{2^{rel_i(c_1)}-1}{log_2(i+1)} + \sum_{i=1}^{k} \frac{2^{rel_i(c_2)}-1}{log_2(i+1)} + \cdots \ldots \ldots .. + \sum_{i=1}^{k} \frac{2^{rel_i(c_m)}-1}{log_2(i+1)} \qquad (7.17)$$

If the top *k* recommended papers belong to user's multiple concepts, then all the relevant values rel_i will be relevant to one of the concepts in the set $\{c_1, c_2, \ldots., c_m\}$ with value 1 (i.e. $\frac{1}{log_2(i+1)}$) for all *k* papers. Then the M_IDCG for the top *k* recommended papers is:

$$M\_IDCG_k = \sum_{i=1}^{k} \frac{1}{log_2(i+1)} \qquad (7.18)$$

Hence, instead of the standard IDCG$_k$ ($i.e.\ IDCG\left(c_j\right)_k = \sum_{i=1}^{n\,(c_j)} \frac{1}{log_2(i+1)}$) we will use our modified M_IDCG$_k$ (i.e. $\mathrm{M\_IDCG}_k = \sum_{i=1}^{k} \frac{1}{log_2(i+1)}$). The standard IDCG performs ideal summation according to only $n\ (c_j)$, which is the total number of relevant papers to concept c_j without any consideration to the other interested concepts, whereas the M_IDCG$_k$ performs ideal summation until k to consider all multiple concepts:

$$M\_NDCG(c_j)_k = \frac{DCG(c_j)_k}{\mathrm{M\_IDCG}_k} \qquad (7.19)$$

If we apply the M_IDCG$_k$ to our previous example (example 1) in section 7.3.2, we will find the following results. The M_NDCG$_k$ for the top 10 recommended papers is:

$$M\_NDCG(c_j)_{10} = \frac{DCG(c_j)_{10}}{\mathrm{M\_IDCG}_{10}}$$

where $DCG(c_j)_{10}$ is:

$$DCG(c_j)_{10} = \sum_{i=1}^{10} \frac{2^{rel_i(c_j)} - 1}{log_2(i+1)}$$

and $\mathrm{M\_IDCG}_{10}$ is:

$$\mathrm{M\_IDCG}_{10} = \sum_{i=1}^{10} \frac{1}{log_2(i+1)}$$

The results will be:

$M\_NDCG\ (c_1)_{10} = 0.564\ \ (\,it\ was = 1\ with\ stantdard\ NDCG(c_1)_{10})$

$M\_NDCG\ (c_2)_{10} = 0.232\ \ (\,it\ was = 0.497\ with\ stantdard\ NDCG(c_2)_{10})$

$M\_NDCG\ (c_3)_{10} = 0.073\ \ (\,it\ was = 0.333\ with\ stantdard\ NDCG(c_3)_{10})$

$M\_NDCG\ (c_4)_{10} = 0.066\ \ (\,it\ was = 0.301\ with\ stantdard\ NDCG(c_4)_{10})$

$M\_NDCG\ (c_5)_{10} = 0.064\ \ (\,it\ was = 0.289\ with\ stantdard\ NDCG(c_5)_{10})$

And the summation of M_NDCGs for these multiple concepts is 1:

$$\mathrm{M\_NDCGs} = \sum_{j=1}^{5} M\_NDCG(c_j)_{10} = 1$$

We can take another example to see the difference if a recommended list does not contain all the user's multiple concepts. Example 2 is similar to example 1, but the concepts c_3 and c_4 and c_5 do not exist in the recommended papers.

Example 2:

Assume that {P_1, P_2, …., P_{10}} are the top 10 retrieved papers (i.e. *k*=10). Where $P_1$is the paper with rank 1. We assume a user has five interested concepts *C*= {c_1, c_2, c_3, c_4, c_5}. Assume the following results are from the classifier:

{P_1, P_2, P_3, P_4} belong to c_1,

{P_5, P_6, P_8} belong to c_2,

{P_7} belong to c_6 (c_6 is not interested by the user),

{P_9} belong to c_7 (c_7 is not interested by the user),

and {P_{10}} belong to c_8 (c_8 is not interested by the user).

The results will be:

$M\_NDCG\ (c_1)_{10} = 0.564\ \ (it\ was = 1\ with\ stantdard\ NDCG(c_1)_{10})$

$M\_NDCG\ (c_2)_{10} = 0.232\ (\ it\ was = 0.497\ with\ stantdard\ NDCG(c_2)_{10})$

$M\_NDCG\ (c_3)_{10} = 0\ \ \ (\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

$M\_NDCG\ (c_4)_{10} = 0\ \ \ (\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

$M\_NDCG\ (c_5)_{10} = 0\ \ (\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

And the summation of M_NDCGs for these multiple concepts is:

$$\text{M\_NDCGs} = \sum_{j=1}^{5} M\_NDCG\left(c_j\right)_{10} = 0.796$$

Which means the recommendation list in example one (with M_NDCGs=1) is better than the recommendation list in example 2 (M_NDCGs = 0.796). We can take a third example to see the difference if a recommended list does not contain all the user's multiple concepts and different ranking than example 2. Example 3 is similar to example 2, but the recommended papers P_4 and P_6 do not belong to any of user's interesting concepts, whereas P_7 and P_9 belong to the user's interesting concepts.

Example 3:

Assume that {P_1, P_2, ….., P_{10}} are the top 10 retrieved papers. We assume a user has five interested concepts *C*={c_1, c_2, c_3, c_4, c_5}. Assume the following results are from the classifier:

{P_1, P_2, P_3, $\mathbf{P_7}$} belong to c_1,

{P_5, P_8, $\mathbf{P_9}$} belong to c_2,

{$\mathbf{P_4}$} belong to c_6 (c_6 is not interested by the user),

{$\mathbf{P_6}$} belong to c_7 (c_7 is not interested by the user),

and {P_{10}} belong to c_8 (c_8 is not interested by the user).

The results will be:

$M\_NDCG\ (c_1)_{10} = 0.542(it\ was = 0.962\ with\ stantdard\ NDCG(c_1)_{10})$

$M\_NDCG\ (c_2)_{10} = 0.221\ (\ it\ was = 0.471\ with\ stantdard\ NDCG(c_2)_{10})$

$M\_NDCG\ (c_3)_{10} = 0\quad(\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

$M\_NDCG\ (c_4)_{10} = 0\quad(\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

$M\_NDCG\ (c_5)_{10} = 0\ \ (\ it\ was = 0\ with\ stantdard\ NDCG(c_3)_{10})$

And the summation of M_NDCGs for these multiple concepts is:

$$\text{M\_NDCGs} = \sum_{j=1}^{5} M\_NDCG\left(c_j\right)_{10} = 0.763$$

Which means the recommendation list in example 2 (with M_NDCGs=0.796) is better than the recommendation list in example 3 (M_NDCGs =0.763). Therefore, the modified M_NDCG_k using the modified M_IDCG_k represents a good measure for a recommender system for multiple concepts, whereas the standard NDCG cannot differentiate good and bad ranking results when we want to measure a system's performance for multiple concepts. Table 7.2 summarize the three examples. The highlighted papers are the related papers to the user's interested concepts. A small arrow for every highlighted paper is drawn to illustrate the concept that is related to the paper. The best recommended list among these examples is the recommended list in example 1, all the ten papers are related to the user's interested concepts. The recommended lists in example 2 and 3 contain seven papers related to user's interesting concepts. However, the ranking in example 2 is better than the ranking in example 3. We added two more examples, example 4 and 5, to the table 7.2. The recommended list in example 4 contains four papers that are related to the user's interested concepts. The worst recommended list among these examples is the recommended list in example 5, it contains only three papers that are related to the user's interested concepts.

Recommended list	M_NDCGs	Standard NDCGs
Example 1: $P_{1\rightarrow c1}$, $P_{2\rightarrow c1}$, $P_{3\rightarrow c1}$, $P_{4\rightarrow c1}$, $P_{5\rightarrow c2}$, $P_{6\rightarrow c2}$, $P_{7\rightarrow c3}$, $P_{8\rightarrow c2}$, $P_{9\rightarrow c4}$, $P_{10\rightarrow c5}$.	1	2.42
Example 2: $P_{1\rightarrow c1}$, $P_{2\rightarrow c1}$, $P_{3\rightarrow c1}$, $P_{4\rightarrow c1}$ $P_{5\rightarrow c2}$, $P_{6\rightarrow c2}$, P_7, $P_{8\rightarrow c2}$, P_9, P_{10}.	0.8	1.5
Example 3: $P_{1\rightarrow c1}$, $P_{2\rightarrow c1}$, $P_{3\rightarrow c1}$, P_4, $P_{5\rightarrow c2}$, P_6, $P_{7\rightarrow c1}$, $P_{8\rightarrow c2}$, $P_{9\rightarrow c2}$, P_{10}.	0.76	1.43
Example 4: $P_{1\rightarrow c1}$, $P_{2\rightarrow c1}$, $P_{3\rightarrow c1}$, $P_{4\rightarrow c2}$, P_5, P_6, P_7, P_8, P_9, P_{10}.	0.56	1.43
Example 5: $P_{1\rightarrow c1}$, $P_{2\rightarrow c1}$, $P_{3\rightarrow c2}$, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}.	0.47	1.5

Table 7.2. Comparing the proposed M_NDCG metric against the standard NDCG metric.

It can be seen that the standard NDCGs metric does not measure the recommended lists accurately. The recommended list in example 2 has been given the value 1.5 with the standard NDCGs metric, which is the same value that is given for the recommended list in example 5. However, in fact the recommended list in example 2 is better than the recommended list in example 5. Moreover, the recommended lists in examples 3 and 4 have been given the same value with NDCGs = 1.43, whereas in fact the recommended list in example 3 is better than the recommended list in example 4. On the other hand, the proposed M_NDCGs metric provides better ranking measure than the standard NDCGs. It shows that the best recommended list is in example 1 with M_NDCGs=1. Then, the second best recommended list is in example 2 with M_NDCGs =0.8. The worst list is the recommended list in example 5 with M_NDCGs=0.47. Therefore, the proposed M_NDCGs metric is able to measure the ranking performance and the quality of the recommended lists more accurately than the standard NDCGs metric.

7.3.4 Evaluation results using the M_NDCGs metric

We compared our hybrid system by using the M_NDCGs metric against our three individual systems: the content-based short-term (CBS) system, the content-based long-term (CBL) system and the collaborative system. We used the same evaluation methodology in section 7.2.1, however instead of using the precision metric, we used the new M_NDCGs metric. Table 7.3 presents the results. The results show that the hybrid system significantly improves the ranking performance. The hybrid system has the highest M_NDCGs of 0.58, while the CBS system scored the

System	**M_NDCGs**
Hybrid System	0.58
CBS System	0.37
CBL System	0.31
Collaborative System	0.26

Table 7.3. The results for the new M_NDCG metric.

second best M_NDCGs of 0.37. Then, the CBL system has M_NDCGs of 0.31. The hybrid system has 21% improvement in the ranking performance comparing with the CBS system and 27% comparing with the CBL system. The collaborative system achieved the lowest M_NDCGs of 0.26. The hybrid system has 32% improvement in the ranking performance comparing with the collaborative system. Therefore, our dynamic hybrid system is able to rank the recommended lists to provide a better recommendation to the users.

7.4 Conclusions

This chapter presented our novel dynamic hybrid system (DHRP) for the research paper domain. The DHRP system uses our previous content-based and collaborative models in chapters 5 and 6. The content-based models are useful to detect the user's current short-term and long-term interests, whereas the collaborative model is employed to discover the user's future interests by involving the similar users' profiles. These models are integrated into the DHRP system by using a mechanism that can be seen as the mean of all recommendation lists. The DHRP model is evaluated using two types of means: arithmetic mean and harmonic mean. Moreover, different values of Υ parameter (the importance factor for the collaborative list) is tested to find the optimal value that provides the best overall performance for our hybrid recommender system. The evaluation results illustrate that the DHRP system has superior performance with harmonic mean rather than arithmetic mean. Then, the DHRP hybrid system with harmonic mean is compared using the precision metric with the individual systems: content-based short-term system, content-based long-term system and collaborative system. The evaluation outcomes demonstrate that the hybrid system is able to provide better recommendation than the individual systems. Moreover, the evaluation results show that the content-based systems provide better results than the collaborative system. This may lead to conclude that the recommendations based on the concepts that are produced by a user's profile itself in the content-based systems are more vital than the recommendations that are produced based on the other user's profiles in the collaborative system. Nonetheless, all these recommendations are integrated together to produce an improved recommendation list in our hybrid recommender system. Furthermore, in this chapter we presented a new ranking metric called M_NDCGs. One of the limitations in the existing metrics is that their measures are according to single user's concept not for multiple user's concepts. This new M_NDCGs metric is an improved version of the standard NDCG metric to evaluate a recommender system for multiple concepts. Our M_NDCGs metric is able to measure the performance of a recommender system for multiple user's concepts at the same time. This new ranking metric shows that the dynamic hybrid system significantly improves the ranking performance comparing with the individual systems. The hybrid system has 21% improvement in the ranking performance comparing with the CBS system, 27% comparing with the CBL system and 32% comparing with the collaborative system. Therefore, our dynamic hybrid system is able to rank effectively the recommended lists to provide better recommendations to the users.

Chapter 8. Conclusions

This final chapter summarises the thesis and its contributions, then make suggestions for potential future work. In section 8.1, the contributions of this research are presented and discussed. In section 8.2, we discuss the limitations of the proposed work and the direction of future work is recommended.

8.1 Summary and thesis contributions

The work presented in this thesis contributes to the development of models and algorithms of recommender systems for the research paper domain. We developed the following models: the content-based DNTC model, the content-based short-term model, the content-based long-term model, the collaborative model for future interests and the hybrid model. Each model is able to solve a specific problem. We evaluated each model in isolation in order to test different settings and parameters to find the optimum performance for each model, as well as provide a direct comparison with other similar models proposed in the literature. Finally, all the models were integrated into one unified hybrid system and its overall performance was measured.

8.1.1 Modelling dynamic user profiles using an ontology

The first contribution of this thesis is modelling of dynamic user profiles that are able to adapt to the changes in multiple user interests and that are compatible with the requirements of advanced ontologies. We use a deep multilevel hierarchal ontology to represent users' interests and build user profiles where each interest is represented as a semantical concept from an existing ontology. In such representation, the objective is to overcome the limitation in the simple weighted keyword method that suffers from the semantic ambiguity. We use the ontology for the 2012 ACM CCS, which is far richer and more complex than the previous 1998 ACM CCS ontology. This ontology is employed in the proposed Dynamic Normalized Tree of Concepts (DNTC) user modelling technique. The user profiling phase creates a user profile as a dynamic normalized tree of concepts. Building a user profile as a tree of concepts maintains the parent-child relationships between the concepts in the ontology. These relationships can be useful while computing the similarity between a user profile and that of a research paper's profile. Normalizing the user's tree of concepts by the number of research papers read by the user provides a more accurate comparison between a research paper profile and a user profile. The user profile is used with a dynamic tree edit distance method to compare the new unseen research papers which are also represented as a tree of concepts. We performed offline evaluations to evaluate the performance of our proposed system. We compared our DNTC system against two baselines: recommender system using the dynamic vector of concepts (DVC) and recommender system using the non-normalized tree of concepts (NNT) (Chandrasekaran et al., 2008). Our results show that our novel DNTC model significantly outperforms both the DVC and the NNT systems. The DVC and the NNT systems are not able to handle the complexity of the users' behaviour. By using the BibSonomy dataset, the evaluation outcomes show that the DNTC system has 28% higher MAP result than the DVC system and 19% higher than the NNT system. Therefore, we can conclude that our novel DNTC system is able to provide high average precision when a user has read a large number of research papers and has a large distribution of multiple concepts.

8.1.2 Modelling short-term and long-term user interests

To the best of our knowledge, the current recommender systems for the research paper domain do not consider short-term and long-term interests. They mostly use the whole user reading history. Existing short-term and long-term user modelling techniques have been developed for domains such as recommending web pages and news articles, where the user reading behaviour is different from that of the research paper domain. These models depend on continuous time-based user behaviour measured in days for the web pages domain or hours in the news domain. In this thesis, we analysed users' reading behaviour within the research paper domain of the BibSonomy dataset that contains real users' records. Our analysis shows that the users' reading of research papers have different durations of inactive days and inactive months which can affect the performance of a recommender system that depends on the continuous time-based method. Moreover, the number of concepts that are involved in users' reading behaviour has a large distribution of concepts. Some of the concepts are short-term concepts that stay less than one month in a user's record. These concepts do not reflect the long-term interests of a user. Therefore, we developed our short-term and long-term models based on our analysis of users' reading behaviours of the research paper domain. The short-term model is based on a novel personalized dynamic sliding window (PDSW) technique where the window length is adapted according to the ratio between the number of concepts and the number of research paper recently read by the user. Therefore, our short-term model is able to change dynamically according to the user's changing reading behaviour. We compared our PDSW short-term model against three systems: the DNTC system, the Static window-time-based model (Gao et al., 2013) and the Dynamic time-based model for the short-term interests (Hawalah and Fasli, 2015). Our evaluations show that our PDSW model achieves MAP of 0.76, which significantly outperforms the baseline systems. The long-term model determines the user's long-term concepts and then selects the research papers that represent those concepts. The user's long-term profile is built from the selected research papers. We compared our long-term model against three systems: the DNTC system, the Time-based forgetting factor model in (Gao et al., 2013) and the Dynamic time-based for long-term interests in (Hawalah and Fasli, 2015). Our long-term model outperforms the baselines and achieves the MAP of 0.81. The performance advantage of our short-term and long-term models is because they can effectively learn different users' reading behaviours implicitly without the need for any intervention from the user. Moreover, they dynamically adapt to the changes in a user's reading behaviour over time.

8.1.3 Predicting user future interests

Predicting user future interests is complex because future interests do not exist in the user's profile. Consequently, there is a need to involve other users' profiles who are similar to the target user by using a collaborative filtering method. Finding similar users in the research paper domain is a complicated task mainly because of the data sparsity problem. For example, in the movie domain, there may be several users who have watched the same movies. Hence, similar users can be found for most users and hence recommendations can be given. However, in the research paper domain, several new research papers have not been read by any user and further, a new user may read only a few research papers. This leads to an inability to successfully locate similar users and hence leads to weak recommendations. Therefore, we developed a novel recommendation model that is able to predict user future interests in the research paper domain. Our novel collaborative method computes the similarity between users whose profiles are represented as Dynamic Normalized Tree of Concepts using the 2012 ACM CCS ontology. Then, a Community-Centric Tree of concepts (CCT) is generated which is used to recommend research papers to the users. We performed offline evaluations using the BibSonomy dataset. Different values for the parameters in our collaborative model are tested to find the optimal values. Then our model is compared with two systems: the content-based DNTC and the User-based Collaborative Filtering (UBCF). Our collaborative system with CCT significantly outperforms the DNTC system and the UBCF system because it maintains the parent-child relationships between the concepts from the 2012 ACM CCS ontology; considers other potential interests that can be extracted from similar users to the target user; and avoids the problem of sparsity. The evaluation results show that our collaborative system has 28% improvement in predicting future interests comparing with the content-based DNTC system and 24% comparing with the UBCF system.

8.1.4 Integrating different types of user interests

User profiles include different types of interests such as current short-term interests, current long-term interests and future interests. We integrated these different types of interests to one unified dynamic hybrid system in order to provide a user with recommendations of research papers that are relevant to user interests at the appropriate time. The dynamic hybrid system integrates our content-based and collaborative models by using a mechanism that can be seen as the mean of all the recommendation lists. The dynamic hybrid model is evaluated using two types of means: arithmetic mean and harmonic mean. Moreover, different values of Υ parameter (the importance factor for the collaborative list) is tested to find the optimal value that provides the best overall performance for our hybrid recommender system. The evaluation results illustrate that the dynamic hybrid system has superior performance with harmonic mean rather than arithmetic mean. Then, the dynamic hybrid system with harmonic mean is compared using the precision metric with the individual systems: content-based short-term system, content-based long-term system and collaborative system. The evaluation outcomes demonstrate that the hybrid system is able to provide better recommendation than the individual systems. The evaluation results for 60 active days shows that the dynamic hybrid system has 14% improvement in recommendation comparing with the content-based short-term system, 21% comparing with the content-based long-term system and 27% comparing with the collaborative system. The performance advantage of our hybrid system is because it can effectively adapt to users' shift or drift of interests during the users' short and long term goals. Moreover, for deeper analysis, the evaluation results show that the content-based systems provide better results than the collaborative system. This may lead to conclude that the recommendations based on the concepts that are produced by a user's profile him/herself in the content-based systems are more vital than the recommendations that are produced based on other user's profiles in the collaborative system. Nonetheless, all these recommendations are integrated together to produce an improved recommendation list in our hybrid recommender system. The results obtained from this evaluation confirmed our argument in this thesis as our hybrid system of modelling and integrating multiple user interests and concepts can bring significant benefits to a recommender system in the research paper domain. That is, modelling of dynamic multi-concept user profiles allowed the dynamic hybrid system to retrieve those research papers that are highly relevant to user interests at the appropriate time.

8.1.5 New ranking measure

One of the main advantages of the dynamic hybrid model is the ranking of the results according to the harmonic mean to provide better recommendations to the users. Therefore, we had the need to measure the improvement in the ranking performance for the dynamic hybrid system against the other systems. However, the existing metrics have a limitation in that their measures are according to single user's concept and not for multiple concepts. Therefore, we proposed a new ranking measure called M_NDCGs. This new metric is an improved version of the standard NDCG metric to evaluate a recommender system which uses multiple concepts. Instead of the standard $IDCG_k$ we use our modified M_IDCG_k. The standard IDCG performs ideal summation according to only one concept without any consideration of multiple concepts, whereas our M_IDCG_k performs ideal summation for multiple concepts. Therefore, the proposed M_NDCGs metric is able to measure the performance of a recommender system for multiple concepts. This new ranking metric shows that the dynamic hybrid system significantly improves the ranking performance compared with the individual systems. The hybrid system has 21% improvement in the ranking performance compared with the content-based short-term system, 27% comparing with the content-based long-term system and 32% comparing with the collaborative system. Therefore, our dynamic hybrid system is able to rank effectively the recommended lists to provide better recommendations to the users.

8.2 Research limitations and future work

In this thesis, we have proposed different models, algorithms and techniques to model multiple dynamic user profiles for recommender systems in the research paper domain. These models, algorithms and techniques covered a wide range of areas including learning and adapting ontological user profiles, capturing content-based short-term and long-term user interests, predicting future interests using the collaborative model, integrating different types of user interests to the dynamic hybrid system and proposing a new ranking measure for multiple concepts. Although a wide range of problems has been covered in this thesis, further improvements of the proposed models, as well as the incorporation of new techniques, can be highlighted. Next, we discuss some of the limitations of our work and recommend some directions for future work.

One of the main limitations of this work is that our system is used only with an ontology for the field of computer and information science, which is the 2012 ACM CCS ontology. The system can be more generic by involving other ontologies from different fields such as medical science and business management. For example, Medical Subject Headings (MeSH) (MeSH, 2015) ontology for medical science. The process of training the classifier for another ontology requires procuration a training set for each concept in the ontology. Therefore, the training sets need to be collected to train the classifier for each concept in an ontology. Thus, more work needs to be done to develop a generic recommender system.

Another limitation of this study is that most of our models and techniques depend on different parameters and settings that need to be pre-optimized. For example, we need to test different values of the weight propagation factor α in order to determine the optimal value to maintain the parent-child relationships between the concepts in the ontological user's tree to provide the best results. Similarly, the processes of learning, adapting and exploiting user profiles for short-term, long-term and future interests have a number of parameters that need to be identified. In this thesis, such parameters have been defined by implementing different experiments using training datasets where the values and settings that provide the optimum results were selected. However, such mechanism is slow and requires conducting a large number of experiments. Nevertheless, when our system is applied in different fields and different ontologies, these settings need to be re-identified over and over again. Therefore, more research needs to be done to develop automated techniques that can optimize our system to any field of application.

Another issue in this thesis is that our system ignores negative feedback from users. Negative feedback is when the user did not read a recommended research paper, then this recommended paper is non-relevant to the user interests. The weight of the concepts that are related to the non-relevant research paper can be reduced with a proper technique to make our system more adaptive to improve the recommendations. It is an important research issue that needs further work.

With regard to our new ranking measure M_NDCG, it requires further improvement because its relevance values are only 0 or 1, and we assumed that a paper can be related to only one concept. As future work, the M_NDCG metric can be improved to include more relevance values in the range [0, 1], where it can be assumed that a paper may relate to more than one concept.

Finally, another limitation identified in this thesis is that the proposed models and techniques have not been tested and implemented in a user study. Running such evaluation for this kind of recommender systems for days and months is too expensive in terms of users' time and effort. Moreover, controlling the experimental parameters render this kind of evaluation difficult to reproduce. Therefore, it is best to run the offline evaluations first to provide evidence that the models and algorithms are able to produce good results. Then, a user study evaluation can be conducted in future work.

References

Abdollahi, B. and Nasraoui, O. (2017) 'Using Explainability for Constrained Matrix Factorization', *In Proceedings of the 11th ACM Conference on Recommender Systems (RecSys2017)*, 79-83.

ACM, 1998 Computing Classification System, Association for Computing Machinery. (1998), [online], Available: http://www.acm.org/about/class/1998.

ACM, Association for Computing Machinery Digital Libraries. (2011), [online], Available: http://dl.acm.org/.

ACM, 2012 Computing Classification System, Association for Computing Machinery. (2012), [online], Available: https://www.acm.org/about/class/2012.

Agarwal, N., Haque, E., Liu, H. and Parsons, L. (2005) 'Research paper recommender systems: A subspace clustering approach' in Advances in Web-Age Information Management. Springer, 475-491.

Agarwal, S., Singhal, A. and Bedi, P. (2012) 'Classification of RSS feed news items using ontology', in *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on*, IEEE, 491-496.

Agarwal, S. and Singhal, A. (2014) 'Handling skewed results in news recommendations by focused analysis of semantic user profiles', in *Optimization, Reliabilty, and Information Technology (ICROIT), 2014 International Conference on*, IEEE, 74-79.

Alhabashneh, O., Iqbal, R., Doctor, F. and Amin, S. (2015) 'Adaptive information retrieval system based on fuzzy profiling', *In 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE),* 1-8.

Al Alshaikh, M., Uchyigit G. and Evans, R. (2017a) 'A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling', *In IEEE Eleventh International Conference on Research Challenges in Information Science (IEEE RCIS 2017),* 200-210.

Al Alshaikh, M., Uchyigit G. and Evans, R. (2017b) 'A Novel Short-term and Long-term User Modelling Technique for a Research Paper Recommender System', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017),* 255-262.

Al Alshaikh, M., Uchyigit G. and Evans, R. (2017c) 'Predicting Future Interests in a Research Paper Recommender System Using a Community Centric Tree of Concepts Model', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017)*, 91-101.

Amazon. (2009), [online], Available: http://www.amazon.co.uk.

Bailey, P., De Vries, A.P., Craswell, N. and Soboroff, I. (2007) 'Overview of the TREC 2007 Enterprise Track'. In TREC.

Beel, J., Gipp, B., Langer, S. and Breitinger, C. (2016) 'Research-paper recommender systems: a literature survey', *International Journal on Digital Libraries*, Springer, 305-338.

Benesty, J., Chen, J., Huang, Y. and Cohen, I. (2006) 'Pearson correlation coefficient', *In Noise reduction in speech processing*, Springer, Berlin Heidelberg, 1-4.

Bennett, P. N., White, R. W., Chu, W., Dumais, S. T., Bailey, P., Borisyuk, F. and Cui, X. (2012) 'Modeling the impact of short-and long-term behavior on search personalization', in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, ACM, 185-194.

Bhatia, N. (2010) 'Survey of nearest neighbor techniques', *arXiv preprint arXiv:1007.0085*.

Bobadilla, J., Ortega, F., Hernando, A. and Gutiérrez, A. (2013) 'Recommender systems survey', *Knowledge-Based Systems*, 46, 109-132.

Breese, J. S., Heckerman, D. and Kadie, C. (1998) 'Empirical analysis of predictive algorithms for collaborative filtering', in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., 43-52.

Carnegie Group and Reuters. (2009), [online], Available: http://www.daviddlewis.com/resources/testcollections/reuters21578/.

Chaffee, J. and Gauch, S. (2000) 'Personal ontologies for web navigation', in *Proceedings of the ninth international conference on Information and knowledge management*, ACM, 227-234.

Challam, V., Gauch, S. and Chandramouli, A. (2007) 'Contextual search using ontology-based user profiles', in *Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*, 612-617.

Chandrasekaran, K., Gauch, S., Lakkaraju, P. and Luong, H. P. (2008) 'Concept-based document recommendations for citeseer authors', in *Adaptive hypermedia and adaptive web-based systems*, Springer, 83-92.

CiteSeer, scientific literature digital library and search engine. (2008), [online], Available: http://citeseer.ist.psu.edu.

CiteSeerX, scientific literature digital library and search engine. (2015), [online], Available: http://citeseerx.ist.psu.edu/index.

De Campos, L. M., Fernández-Luna, J. M., Huete, J. F. and Rueda-Morales, M. A. (2010) 'Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks', *International Journal of Approximate Reasoning*, 51(7), 785-799.

Deshpande, M. and Karypis, G. (2004) 'Item-based top-n recommendation algorithms', *ACM Transactions on Information Systems (TOIS)*, 143-177.

Dillon, M. (1983) 'Introduction to modern information retrieval: G. Salton and M. McGill. McGraw-Hill, New York (1983). ISBN 0-07-054484-0'.

Ding, Y. and Li, X. (2005) 'Time weight collaborative filtering', in *Proceedings of the 14th ACM international conference on Information and knowledge management*, ACM, 485-492.

Dong, R., Tokarchuk, L. and Ma, A. (2009) 'Digging Friendship: Paper Recommendation in Social Network', in *Proceedings of Networking & Electronic Commerce Research Conference (NAEC 2009)*, 21-28.

Dumais, S. T., Furnas, G. W., Landauer, T. K., Deerwester, S. and Harshman, R. (1988) 'Using latent semantic analysis to improve access to textual information', in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 281-285.

Ekstrand, M.D., Riedl, J.T. and Konstan, J.A. (2011) 'Collaborative filtering recommender systems', *Foundations and Trends in Human–Computer Interaction*. 81-173.

Farlex, Inc. (2016) 'concept', [online], Available: http://www.thefreedictionary.com/concept.

Fouss, F., Pirotte, A., Renders, J.M. and Saerens, M. (2007) 'Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation', *IEEE Transactions on knowledge and data engineering*, 19(3), 355-369.

Fanaee-T, H. and Yazdi, M. (2011) 'A Novel Ontology-based Recommender System for Online Forums', in *Proceedings of the 3rd IEEE International Conference on Information Management and Engineering*, 190-197.

Gao, Q., Xi, S.M. and Im Cho, Y., (2013) 'A multi-agent personalized ontology profile based user preference profile construction method'. *In 44th IEEE International Symposium on Robotics (ISR)*, 1-4.

Gauch, S., Speretta, M., Chandramouli, A. and Micarelli, A. (2007) 'User profiles for personalized information access', *In the adaptive web.* Springer, 54-89.

Gauch, S., Ravindran, D. and Chandramouli, A. (2010) 'KeyConcept: Conceptual Search and Pruning Exploiting Concept Relationships', *Journal of Intelligent Systems*, 19(3), 265-288.

Gori, M., Pucci, A., Roma, V. and Siena, I. (2007) 'ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines', *In IJCAI*, Vol. 7, 2766-2771.

GroupLens Research Project. (2005), [online], Available: http://www.grouplens.org/.

Gruber, T. R. (1993) 'A translation approach to portable ontology specifications', *Knowledge acquisition*, 5(2), 199-220.

Gupta, K. M., Aha, D. W., Marsh, E. and Maney, T. (2002) 'An architecture for engineering sublanguage WordNets', *in Proceedings of the First International Conference on Global WordNet*, 207-215.

Hawalah, A. and Fasli, M. (2011) 'A multi-agent system using ontological user profiles for dynamic user modelling', in *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*, IEEE Computer Society, 430-437.

Hawalah, A. and Fasli, M. (2015) 'Dynamic user profiles for web personalisation', *Expert Systems with Applications*, 42(5), 2547-2569.

IEEE digital libraries. (2011), [online], Available: http://ieeexplore.ieee.org/Xplore/home.jsp.

Isinkaye, F.O., Folajimi, Y.O. and Ojokoh, B.A. (2015) 'Recommendation systems: Principles, methods and evaluation', *Egyptian Informatics Journal*, 261-273.

Jack, K. (2013) 'What makes a Search Engine different from a Recommender System?', [online], Available: https://krisjack.wordpress.com/2013/10/01/what-makes-a-search-engine-different-from-a-recommender-system/.

Jain, M., (2012) 'Algorithms for Research Paper Recommendation System', *International Journal of Information Technology*, 5(2), 443-445.

Jannach, D., Zanker, M., Felfernig, A. and Friedrich, G. (2010) *Recommender systems: an introduction*, Cambridge University Press.

Jomsri, P., Sanguansintukul, S. and Choochaiwattana, W. (2010) 'A framework for tag-based research paper recommender system: an IR approach', in *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*, IEEE, 103-108.

Kacem, A., Boughanem, M. and Faiz, R. (2014) 'Time-sensitive user profile for optimizing search personlization' in *User Modeling, Adaptation, and Personalization*Springer, 111-121.

Khan, L. and Luo, F. (2002) 'Ontology construction for information selection', in *Tools with Artificial Intelligence, 2002. (ICTAI 2002). Proceedings. 14th IEEE International Conference on*, IEEE, 122-127.

Knowledge and Data Engineering Group, University of Kassel: Benchmark Folksonomy Data from BibSonomy, version of January 1st, 2017.

Kodakateri Pudhiyaveetil, A., Gauch, S., Luong, H. and Eno, J. (2009) 'Conceptual recommender system for CiteSeerX', in *Proceedings of the third ACM conference on Recommender systems*, ACM, 241-244.

Kotkov, D., Veijalainen, J. and Wang, S. (2016) 'Challenges of serendipity in recommender systems', *In Proceedings of the 12th International conference on web information systems and technologies.* 23-41

Kritikou, Y., Demestichas, P., Adamopoulou, E., Demestichas, K., Theologou, M. and Paradia, M. (2008) 'User Profile Modeling in the context of web-based learning management systems', *Journal of Network and Computer Applications*, 31(4), 603-627.

Lakkaraju, P., Gauch, S. and Speretta, M. (2008) 'Document similarity based on concept tree distance', in *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, ACM, 127-132.

Lee, J., Lee, K. and Kim, J. G. (2013) 'Personalized academic research paper recommendation system', *arXiv preprint arXiv:1304.5457*.

Leung, K. W.-T. and Lee, D. L. (2010) 'Deriving concept-based user profiles from search engine logs', *Knowledge and Data Engineering, IEEE Transactions on*, 22(7), 969-982.

Li, L., Yang, Z., Wang, B. and Kitsuregawa, M. (2007) 'Dynamic adaptation strategies for long-term and short-term user profile to personalize search' in *Advances in Data and Web Management*Springer, 228-240.

Li, L., Zheng, L., Yang, F. and Li, T. (2014) 'Modeling and broadening temporal user interest in personalized news recommendation', *Expert Systems with Applications*, 41(7), 3168-3177.

Lu, J., Wu, D., Mao, M., Wang, W. and Zhang, G. (2015) 'Recommender system application developments: a survey', *Decision Support Systems*, 74, 12-32.

Ma, H., King, I. and Lyu, M.R. (2007) 'Effective missing data prediction for collaborative filtering', *In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 39-46).

Manning, C. D., Raghavan, P. and Schütze, H. (2008) *Introduction to information retrieval*, Cambridge university press Cambridge.

McNee, S. M., Kapoor, N. and Konstan, J. A. (2006) 'Don't look stupid: avoiding pitfalls when recommending research papers', in *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, ACM, 171-180.

McSherry, F. and Najork, M. (2008) 'Computing information retrieval performance measures efficiently in the presence of tied scores' in *Advances in information retrieval*, Springer, 414-421.

Mendeley. (2014), [online], Available: http://www.mendeley.com/.

MeSH, Medical Subject Headings. (2015), [online], Available: https://www.nlm.nih.gov/pubs/factsheets/mesh.html.

Moghaddam, S., Jamali, M. and Ester, M. (2012) 'ETF: extended tensor factorization model for personalizing prediction of review helpfulness', in *Proceedings of the fifth ACM international conference on Web search and data mining*, ACM, 163-172.

MovieLens. (2005), [online], Available: http://movielens.umn.edu/login.

Mnih, A. and Salakhutdinov, R.R. (2008) 'Probabilistic matrix factorization', *In Advances in neural information processing systems*, 1257-1264.

Nadee, W., Li, Y. and Xu, Y. (2013) 'Acquiring user information needs for recommender systems', *In Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, 5-8.

Nanas, N., Vavalis, M. and Kellis, L. (2009) 'August. Immune Learning in a Dynamic Information Environment', *In ICARIS*. 192-205.

Netflix, (2014), [online], Available: https://www.netflix.com.

ODP, Open Directory Project, (2011), [online], Available: http://www.dmoz.org/.

Oh, K.-J., Lee, W.-J., Lim, C.-G. and Choi, H.-J. (2014) 'Personalized news recommendation using classified keywords to capture user preference', in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, IEEE, 1283-1287.

OWL, Web Ontology Language, (2013), [online], Available: https://www.w3.org/TR/owl-ref/.

Park, D. H., Kim, H. K., Choi, I. Y. and Kim, J. K. (2012) 'A literature review and classification of recommender systems research', *Expert Systems with Applications*, 39(11), 10059-10072.

Quadrana, M., Karatzoglou, A., Hidasi, B. and Cremonesi, P. (2017) 'Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks', *In Proceedings of 11th ACM Conference on Recommender Systems (RecSys2017)*, 130-137.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. (1994) 'GroupLens: an open architecture for collaborative filtering of netnews', in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, ACM, 175-186.

Ricardo, B. and Berthier, R. (2011) 'Modern Information Retrieval the concepts and technology behind search second edition', *Addision Wesley*, 84, 2.

Ricci, F., Rokach, L. and Shapira, B. (2011) *Introduction to recommender systems* *handbook*, Springer.

Schafer, J. B., Frankowski, D., Herlocker, J. and Sen, S. (2007) 'Collaborative filtering recommender systems' in *the adaptive web*Springer, 291-324.

Science Paper. (2012), [online], Available: http://www.paper.edu.cn/en.

Shani, G. and Gunawardana, A. (2011) 'Evaluating recommendation systems' in *Recommender systems handbook*Springer, 257-297.

Shi, Y., Larson, M. and 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), 3.

Singh, A. P. and Gordon, G. J. (2008) 'Relational learning via collective matrix factorization', in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 650-658.

Soboroff, I. and Nicholas, C. (2005) 'Combining content and collaboration in text filtering', in *Proceedings of the IJCAI*, 86-91.

Sparck, K., and Willett, P. (1997) 'Readings in Information Retrieval'. Morgan Kaufmann, San Mateo, US.

Sánchez, D. M., Cavero, J. M. and Martínez, E. M. (2007) 'The road toward ontologies' in *Ontologies*, Springer, 3-20.

Sugiyama, K. and Kan, M.Y. (2010) 'Scholarly paper recommendation via user's recent research interests', *In Proceedings of the 10th ACM annual joint conference on Digital libraries*, 29-38.

Tamine-Lechani, L., Boughanem, M. and Zemirli, N. (2008) 'Personalized document ranking: Exploiting evidence from multiple user interests for profiling and retrieval', *JDIM*, 6(5), 354-365.

Tang, X. and Zeng, Q. (2012) 'Keyword clustering for user interest profiling refinement within paper recommender systems', *Journal of Systems and Software*, 85(1), 87-101.

Tarakci, H. and Cicekli, N. K. (2014) 'A Formal Framework for Hypergraph-Based User Profiles' in *Information Sciences and Systems 2014*, Springer, 285-293.

Vassiliou, C., Stamoulis, D., Martakos, D. and Athanassopoulos, S. (2006) 'A recommender system framework combining neural networks & collaborative filtering', in *Proceedings of the 5th WSEAS international conference on Instrumentation, measurement, circuits and systems*, World Scientific and Engineering Academy and Society (WSEAS), 285-290.

Wang, Y., Wang, L., Li, Y., He, D., Chen, W. and Liu, T.-Y. (2013) 'A theoretical analysis of NDCG ranking measures', in *Proceedings of the 26th Annual Conference on Learning Theory (COLT 2013)*,156-167.

Yao, Y. (1995) 'Measuring retrieval effectiveness based on user preference of documents', *Journal of the American Society for Information Science*, 46(2), 133-138.

Yang, C., Wei, B., Wu, J., Zhang, Y. and Zhang, L. (2009) 'CARES: a ranking-oriented CADAL recommender system', *In Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, 203-212.

Yang, W., Tang, R. and Lu, L. (2016) 'A fused method for news recommendation', *In IEEE International Conference in Big Data and Smart Computing (BigComp),* 341-344.

Ye, N., Gauch, S., Wang, Q. and Luong, H. (2010) 'An Adaptive Ontology based Hierarchical Browsing System for CiteSeerx', in *Knowledge and Systems Engineering (KSE), 2010 Second International Conference on*, IEEE, 203-208.

Zeb, M. A. and Fasli, M. (2011) 'Adaptive user profiling for deviating user interests', in *Computer Science and Electronic Engineering Conference (CEEC), 2011 3rd*, IEEE, 65-70.

Zeb, M.A. and Fasli, M. (2012) 'Dynamically Adaptive User Profiling for Personalized Recommendations', *In IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT),* 604-611.

Zhang, R., Liu, Q.D., Gui, C., Wei, J.X. and Ma, H. (2014) 'Collaborative filtering for recommender systems', *In 2014 IEEE Second International Conference on Advanced Cloud and Big Data (CBD),* 301-308.

Zhu, X., Goldberg, A. B., Van Gael, J. and Andrzejewski, D. (2007) 'Improving Diversity in Ranking using Absorbing Random Walks', in *HLT-NAACL*, 97-104.

Appendix A: Templets for users' scenarios.

a) Users' template scenarios 1, 2 and 3 consider <u>three concepts</u>:

Scenario 1: small quantity of research papers (15)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	4	6-9
Concept 3	6	10-15

Scenario 2: medium quantity of research papers (30)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	10	1-10
Concept 2	11	11-21
Concept 3	9	22-30

Scenario 3: large quantity of research papers (50)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	15	1-15
Concept 2	18	16-33
Concept 3	17	34-50

b) Users' template scenarios 4, 5 and 6 consider <u>four concepts</u>:

Scenario 4: small quantity of research papers (15)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	5	11-15

Scenario 5: medium quantity of research papers (30)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	6	1-6
Concept 2	9	7-15
Concept 3	7	16-22
Concept 4	8	23-30

Scenario 6: large quantity of research papers (50)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	14	1-14
Concept 2	11	15-25
Concept 3	13	26-38
Concept 4	12	39-50

c) Users' template scenarios 7, 8, and 9 consider five concepts:

Scenario 7: small quantity of research papers (15)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	2	11-12
Concept 5	3	13-15

Scenario 8: medium quantity of research papers (30)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	7	6-12
Concept 3	6	13-18
Concept 4	4	19-22
Concept 5	8	23-30

Scenario 9: large quantity of research papers (50)

Concepts	Number of research papers	Time sequence for user reading
Concept 1	9	1-9
Concept 2	11	10-20
Concept 3	10	21-30
Concept 4	12	31-42
Concept 5	8	43-50

Appendix B: The detailed results for the PDSW short-term model against three short-term systems.

	AVG P			
Active Day	PDSW	DNTC	Dynamic time-based	Static window-time-based
2	0.48	0.47	0.49	0.42
3	0.53	0.5	0.52	0.44
4	0.69	0.53	0.54	0.46
5	0.72	0.56	0.52	0.43
6	0.7	0.54	0.53	0.405
7	0.75	0.53	0.5	0.325
8	0.78	0.52	0.7	0.375
9	0.8	0.45	0.61	0.335
10	0.66	0.5	0.51	0.325
11	0.75	0.39	0.415	0.345
12	0.7	0.42	0.445	0.365
13	0.75	0.45	0.465	0.385
14	0.72	0.48	0.445	0.355
16	0.75	0.46	0.455	0.405
17	0.8	0.35	0.425	0.325
18	0.74	0.52	0.625	0.375
19	0.76	0.45	0.535	0.335
20	0.75	0.42	0.625	0.425
21	0.76	0.48	0.605	0.525
22	0.85	0.6	0.655	0.615
23	0.84	0.46	0.635	0.605
24	0.82	0.54	0.565	0.585
25	0.87	0.3	0.605	0.625
26	0.78	0.53	0.555	0.575
27	0.74	0.45	0.425	0.465
28	0.73	0.2	0.525	0.455
29	0.77	0.47	0.495	0.525
30	0.7	0.5	0.535	0.525

31	0.67	0.62	0.645	0.555
32	0.73	0.52	0.605	0.575
33	0.8	0.38	0.565	0.535
34	0.75	0.52	0.625	0.585
35	0.66	0.49	0.525	0.545
36	0.74	0.52	0.355	0.495
37	0.71	0.45	0.565	0.525
38	0.79	0.53	0.625	0.595
39	0.8	0.57	0.615	0.425
40	0.83	0.28	0.655	0.615
41	0.75	0.53	0.575	0.555
42	0.85	0.57	0.665	0.625
43	0.8	0.44	0.595	0.525
44	0.86	0.36	0.475	0.545
45	0.8	0.5	0.535	0.575
46	0.79	0.37	0.505	0.485
47	0.71	0.52	0.565	0.505
48	0.82	0.57	0.565	0.595
49	0.8	0.55	0.615	0.565
50	0.84	0.43	0.545	0.525
51	0.8	0.48	0.595	0.555
52	0.85	0.32	0.575	0.605
53	0.73	0.4	0.525	0.485
54	0.8	0.42	0.565	0.495
55	0.8	0.47	0.625	0.545
56	0.85	0.52	0.495	0.495
57	0.83	0.6	0.565	0.525
58	0.78	0.28	0.535	0.425
59	0.8	0.52	0.625	0.565
60	0.84	0.49	0.575	0.525
MAP	**0.76**	**0.47**	**0.55**	**0.49**

Appendix C: Detailed table for comparing the MAP for our hybrid system against our individual systems.

	MAP			
Active Day	**CBS system**	**CBL system**	**Collaborative system**	**Hybrid system**
2	0.1	0.02	0.04	0.07
3	0.17	0.05	0.09	0.12
4	0.22	0.09	0.13	0.19
5	0.3	0.13	0.21	0.28
6	0.32	0.18	0.26	0.3
7	0.33	0.19	0.27	0.35
8	0.34	0.22	0.29	0.37
9	0.39	0.23	0.3	0.42
10	0.4	0.25	0.3	0.45
11	0.39	0.26	0.31	0.46
12	0.41	0.27	0.34	0.52
13	0.45	0.21	0.37	0.57
14	0.49	0.3	0.4	0.56
15	0.42	0.33	0.35	0.55
16	0.51	0.38	0.33	0.6
17	0.57	0.45	0.36	0.62
18	0.4	0.52	0.39	0.64
19	0.53	0.41	0.4	0.68
20	0.6	0.43	0.38	0.63
21	0.55	0.62	0.4	0.65
22	0.58	0.54	0.42	0.69
23	0.51	0.47	0.43	0.66
24	0.57	0.39	0.45	0.67
25	0.46	0.44	0.39	0.69
26	0.5	0.39	0.38	0.7
27	0.52	0.43	0.35	0.71
28	0.57	0.4	0.37	0.7
29	0.51	0.48	0.36	0.73
30	0.46	0.55	0.35	0.75

31	0.44	0.51	0.37	0.74
32	0.53	0.58	0.4	0.76
33	0.46	0.6	0.36	0.75
34	0.45	0.54	0.37	0.72
35	0.48	0.5	0.38	0.7
36	0.53	0.56	0.41	0.71
37	0.57	0.5	0.34	0.7
38	0.5	0.5	0.33	0.66
39	0.54	0.58	0.35	0.69
40	0.55	0.51	0.32	0.67
41	0.46	0.57	0.37	0.7
42	0.51	0.53	0.32	0.72
43	0.57	0.59	0.35	0.71
44	0.58	0.46	0.38	0.73
45	0.51	0.47	0.42	0.71
46	0.53	0.45	0.43	0.7
47	0.55	0.42	0.4	0.72
48	0.5	0.43	0.38	0.73
49	0.52	0.46	0.39	0.74
50	0.5	0.45	0.37	0.73
51	0.55	0.41	0.35	0.74
52	0.48	0.4	0.35	0.71
53	0.51	0.43	0.33	0.7
54	0.5	0.45	0.37	0.72
55	0.53	0.41	0.39	0.71
56	0.5	0.43	0.35	0.73
57	0.56	0.46	0.39	0.74
58	0.54	0.45	0.36	0.72
59	0.55	0.44	0.38	0.73
60	0.56	0.43	0.37	0.72
AVG_MAP	**0.48**	**0.41**	**0.35**	**0.62**

Appendix D: Conference papers resulting from the thesis.

1- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling', *In IEEE Eleventh International Conference on Research Challenges in Information Science (IEEE RCIS 2017)*, 200-210.
(This publication is related to chapter 4).

2- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'A Novel Short-term and Long-term User Modelling Technique for a Research Paper Recommender System', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017)*, 255-262.
(This publication is related to chapter 5).

3- Al Alshaikh, M., Uchyigit G. and Evans, R. (2017) 'Predicting Future Interests in a Research Paper Recommender System Using a Community Centric Tree of Concepts Model', *In the 9th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2017)*, 91-101.
(This publication is related to chapter 6).

A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling

Modhi Al Alshaikh, Gulden Uchyigit, Roger Evans
School of Computing, Engineering and Mathematics
University of Brighton
Brighton, United Kingdom
{m.alalshaikh, g.uchyigit, r.p.evans}@brighton.ac.uk

***Abstract*— The enormous growth of information on the Internet makes finding information challenging and time consuming. Recommender systems provide a solution to this problem by automatically capturing user interests and recommending related information the user may also find interesting. In this paper, we present a novel recommender system for the research paper domain using a Dynamic Normalized Tree of Concepts (DNTC) model. Our system improves existing vector and tree of concepts models to be adaptable with a complex ontology and a large number of papers. The proposed system uses the 2012 version of the ACM Computing Classification System (CCS) ontology. This ontology has a much deeper structure than previous versions, which makes it challenging for previous ontology-based approaches to recommender systems. We performed offline evaluations using papers provided by ACM digital library for classifier training, and papers provided by CiteSeerX digital library for measuring the performance of the proposed DNTC model. Our evaluation results show that the novel DNTC model significantly outperforms the other two models: non-normalized tree of concepts and the vector of concepts models. Further, our DNTC model provides high average precision and reliable results when used in a context which the user has multiple interests and reads a large quantity of papers over time.**

Keywords*— *normalized tree of concepts; recommander system; personalization; user profile; 2012 ACM CCS ontology.

I. INTRODUCTION

The enormous growth of information on the Internet makes finding information both challenging and time consuming. Traditional search engines require the user to manually enter keywords in order to search for relevant web pages or data collections. The results of the search query are displayed to the user based on the order of relevance to the keywords. One of the problems with traditional keyword based search engines is that the user may find it difficult to find the search keywords which will return the best results, especially if the user is searching for information in a new domain. Recommender systems provide a solution to this problem by automatically capturing user interests/preferences and recommending related information the user may also find interesting. There are two ways in which recommender systems are able to capture user preferences: *explicitly*, by enabling the user to enter their preferences, or *implicitly,* by monitoring the user's activities such as browsing the web or reading documents. Collected preferences are stored in a user profile. New items (e.g. documents) are then compared with the user profile and those items which are sufficiently similar are recommended to the user. Existing recommender systems offer efficient personalized services in variety of domains such as movies, music, television, books, documents, e-learning and e-commerce [1].

One of the interesting systems in the document domain is a research paper recommender system. Current research paper recommender systems suffer from a number of limitations that may constrain their recommendation services. One critical limitation in these systems is that they are not compatible with the new advanced ontologies, that have become bigger, more complex and with deeper levels. For example, the 2012 ACM Computing Classification System (CCS) [2] relies on a semantic vocabulary as the source of categories and concepts that reflect the state of the art in the computing discipline. It replaces the previous 1998 version of the ACM CCS (the '98 ACM CSS'), which has served as the de facto standard classification system for the computing field, and has been used by several recent recommender systems (e.g. [3], [8], [9]). The 98 ACM CCS ontology has a three-level hierarchical set of concepts that contains in total 369 concepts [8]. However, to reflect the rapidly developing field of computing research the 98 ACM CCS ontology was updated to the 2012 ACM CCS to include the new deeper level concepts. The 2012 ACM CCS ontology has a poly-hierarchical ontology and maintains a six-level hierarchical tree with more than one thousand concepts [2].

While ontologies are growing bigger and more complex, finding relevant papers related to users' interests becomes a challenging task for the recommender systems. Often dynamic recommender systems use an ontology to create the user's profile as vector of concepts [4]. However, representing the user profile as a vector of concepts assumes that the concepts are independent from each other and does not accurately represent the user's interests. With a complex ontology, there is a need to employ more sophisticated techniques to build a user profile. The tree of concepts model [3] used in conjunction with complex ontologies addresses this problem, but it is static, in that it is unable to dynamically capture new and multiple user's interests. Furthermore, it does not normalize the user model according to the number of papers that are involved in the user profile, which causes its performance to decline significantly if the profile contains larger numbers of papers.

In this paper, we propose a content based recommender system for research papers which addresses these problems. In the proposed system, a user profile is built as a Dynamic Normalized Tree of Concepts (DNTC) by monitoring the user's reading behavior over time. In our DNTC user modelling approach, the parent-child relationships between the concepts from the ontology are maintained whilst computing the similarity between a user profile and the new research papers to be recommended. The DNTC user profile is constructed using the 2012 ACM CCS as a reference ontology. In our offline evaluations we compare the DNTC system with two models: dynamic vector of concepts (DVC) model and non-normalized tree of concepts (NNT) model. We show that our model's performance is equal to or better than previous systems across a range of usage scenarios, and in particular that it is significantly better for the more demanding scenarios (more concepts, more papers) that we are using in our current work on modelling short and long term preferences in recommender systems.

The rest of this paper is organized as follows. Section II presents the related work. Section III discusses our DNTC system. Section IV shows offline evaluation results. Finally, conclusions and future work are given in Section V.

II. RELATED WORK

The accurate representation of user interests and preferences in the form of a user profile is key to the effectiveness of recommender systems [4]. A common profile representation technique is to use weighted feature vectors. We focus on user profiling for document recommendation such as research papers, news and web pages. In this domain, the features can be in the form of keywords automatically extracted from text documents which the user has implicitly or explicitly shown a preference towards. In such representation techniques the keywords are associated with weights to represent the significance of the keyword in the user profile. Lee et al. [5] developed a system that extracts keywords from the user's previously published papers, assuming that the user will be interested in similar papers to their previous research topics. Zeb and Fasli [17] proposed a technique that constructs probabilistic user profiles by subscribing to an RSS (Rich Site Summary) news aggregator. The probabilistic user model is constructed based on implicit user feedback (click response) over a period of time.

A major shortcoming of keyword based user profile representation techniques is that they are not suitable for representation of complex user profiles [4]. This is because representing user interests as simple keywords increases the ambiguity as it lacks semantic information. One way of semantically enriching the user profile representation is through the use of abstract concepts drawn from an ontology instead of words. By mapping between words and concepts in a reference ontology it is possible to build more robust user profiles with reduced user feedback and monitoring. Examples of ontologies that are used in recommender systems include the Open Directory Project (ODP)[1] and the ACM CCS[2]. Such approaches have been shown to provide a significant improvement in the performance of user profiling models in recommender systems [4]. Gauch et al. [4] noted that most of researchers who used ontologies for user profile representation use them in a similar way to weighted keywords in that the concepts are represented as vectors of weighted features, but the features represent concepts rather than words. For example, Agarwal and Singhal [6] employed OWL (Web Ontology Language)[3] to build the user profile. Their system periodically gathers visited news pages by using unique features of RSS feed news items and arranges them in chronological order. The user profile consists of concepts that are interesting to the user. Concepts are given weights based on number of clicks in a session, recency of the session and active session duration. Tang and Zeng [7] used an ontology that is defined by the Sciencepaper Online[4]. Fig.1 shows the concepts of the subject "computer science" in this ontology. The user profile model in [7] computes weights of concepts for the papers that are read by the user and represents them as vectors of concepts. Kodakateri et al. [8] designed a recommender system that recommends potential research papers of interest to users from the CiteSeer database. The 98 ACM CCS is used as a reference ontology. They developed a dynamic user profile that is updated each time the user visits a new research paper.

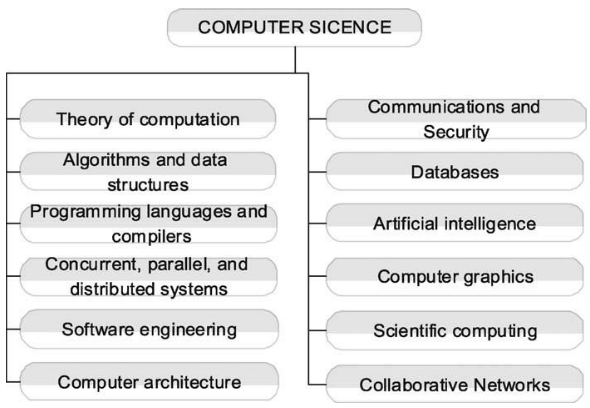

Fig. 1. The classification of "computer science" in the Sciencepaper ontology [7].

[1] http://www.dmoz.org/docs/en/about.html
[2] http://www.acm.org/about/class/
[3] https://www.w3.org/TR/owl-ref/
[4] http://www.paper.edu.cn/en

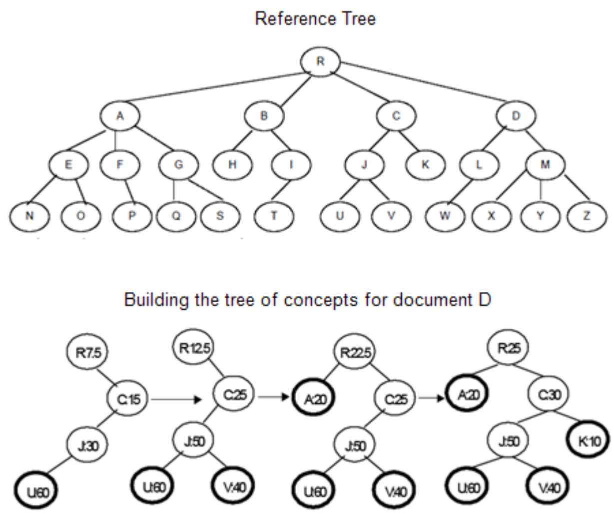

Fig. 2. Tree of Concepts Technique [9].

The concept vectors technique may be sufficient with a simple ontology that consists of two levels of classification, such as primary and secondary subjects as shown in Fig.1. However, with complex ontologies such as ACM CCS ontology that maintains multiple level hierarchies, there is a need to employ more sophisticated techniques to build a user profile. An interesting technique is developed in [3] which represents a user profile as a tree of concepts. Their recommender system is for the research paper domain using the 98 ACM CCS ontology. A user profile is created based on the user's previously published papers. The tree of concepts is created as follows. A paper is submitted to a classifier to determine the list of top concepts with the highest weights. For example, a document D (in Fig.2) have the following concept vector (conceptID, weight):

D = { (U, 60), (V, 40), (A, 20), (K, 10) }

Then, the concept vector is input to the Tree Builder Module [9] to create a (weighted) tree of concepts for the document D based on a reference tree as in Fig. 2. The output tree is a subtree of the reference tree spanning all the concepts in the concept vector. Weights are assigned to leaf nodes from the input vector, and then percolated upwards, reducing by 50% on each step (see Fig. 2). The user profile is constructed by combining the trees of concepts from the user's publications.

The concept vector technique assumes that the elements of the vectors being compared are independent, which is not an accurate representation of the user's preferences [3]. In order to exploit the relationships between the concepts it is more efficient to use the tree of concepts technique, because it can exploit inter-relationships between the concepts through the ontology [3]. However, their user profiling model using the tree of concepts technique is static over time, whereas user preferences and needs are not static but they usually change over time. Moreover, this user profiling technique does not normalize the concept weights. Without normalization, the weights in the user's tree of concepts profile representation are too big to compare accurately with the weights in a tree of concepts for a paper in the recommendation phase. To overcome these problems, we have developed a Dynamic Normalized Tree of Concepts (DNTC) model for user profiles, which we introduce in the next section.

III. Our System

The DNTC recommender system consists of three main phases: papers classification phase, DNTC user profiling phase and recommendation phase. The first phase is responsible for preparing papers and classifying them. The second phase is responsible for tracking user reading activities for papers, and using the papers read by the user to build a user profile as a dynamic normalized tree of concepts. The third phase is recommendation phase that uses a dynamic tree edit distance technique to recommend a set of papers to the user that match his/her preferences. Fig. 3 shows an overview diagram for the proposed system. The next subsection discusses our ontology model. After that, our system's phases will be explained in detail in subsections III.B, III.C and III.D.

A. *Ontology in our system*

In our system, a reference ontology is used for three main purposes:

1) Mapping a paper to the correct concepts using a classification algorithm.
2) Representing a paper profile as a tree of concepts.
3) Representing a user profile as a normalized tree of concepts.

We use the ontology for the 2012 ACM CCS. It relies on a semantic vocabulary as the source of categories and concepts that reflect the state of the art of the computing discipline and is receptive to structural change as it evolves in the future. The usage of the 2012 ACM CCS in our classification phase is explained in the next section.

B. *Research papers classification phase*

In the first phase in our system, we build a classifier using the ACM training dataset and classify a set of research papers from the CiteSeerX [18] dataset (see section IV for more information about these datasets) to the reference ontology. All the papers in the dataset are mapped to the reference ontology by classifying each paper to the correct concepts using the Term Frequency-Inverse Document frequency (TF-IDF) technique [16] and cosine similarity [13]. The classification process consists of two phases:

1) Training phase: During this phase, papers in the ACM training set, which are pre-assigned to one or more concepts in the reference ontology, are used to learn a vector of features for each concept in the ontology.

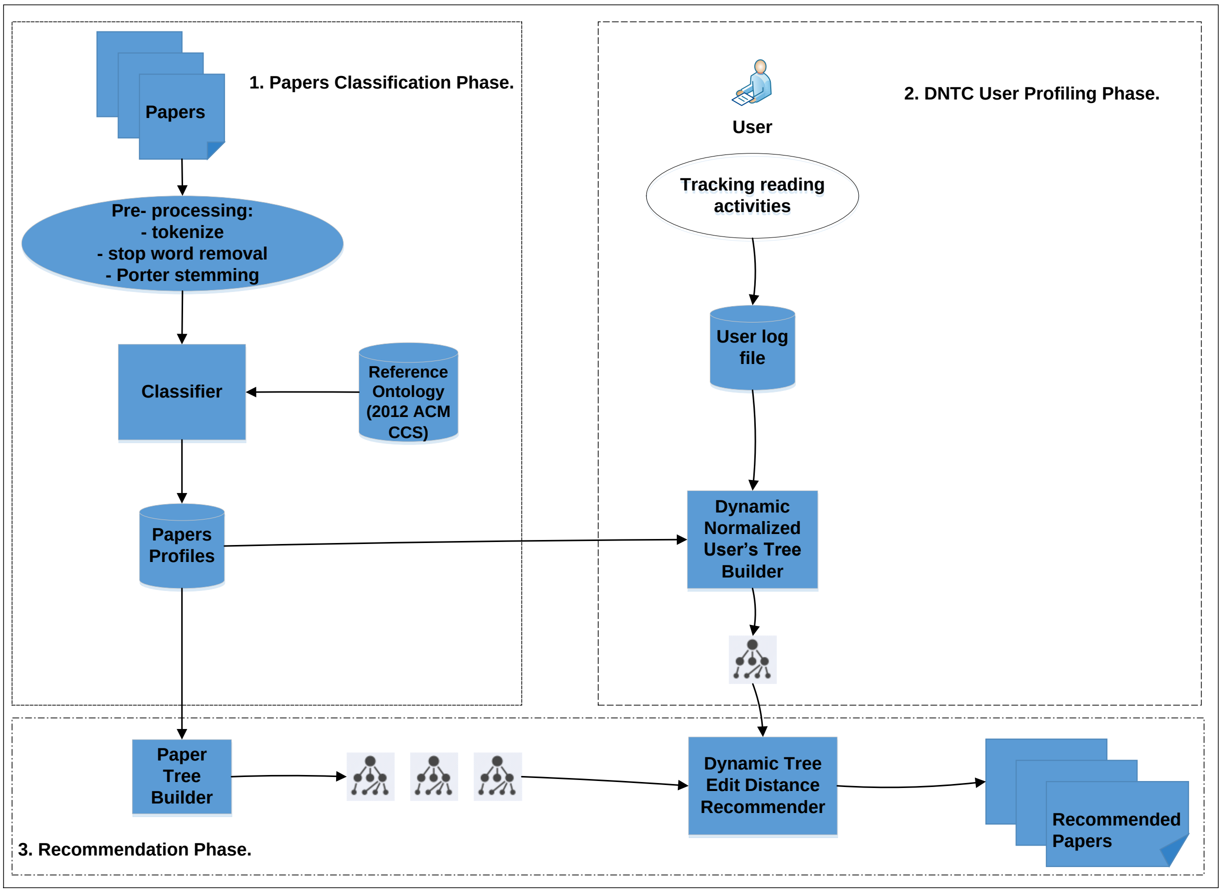

Fig. 3. The DNTC system architecture.

2) Classification phase: In this phase the cosine similarity classifier uses the vectors learnt in the training phase to classify papers in the CiteSeerX dataset. The output is a list of concepts for each input paper along with their corresponding weights which indicate the degree of association between the concept and the paper. The top N concepts for each paper are retained and stored in the research paper profile.

B.1. Training phase

The training set was provided by ACM[1]. The training set contains papers which are pre-assigned to one or more concepts in the 2012 ACM CCS ontology manually by the authors of the papers. Hence, the concepts in the ontology reference are associated with training papers that represents each concept. We combine all the papers to one document (d_j) to represent a concept (c_j). Each document is tokenized and represented as a set of terms constructed from the papers' title, keywords and abstract. We applied some heuristics functions to pre-process the text, these functions are stop words removal and then Porter stemming algorithm [10] which reduces each word (term) to its shortest stem. The documents are then represented as weighted feature vectors by using the TF-IDF weighting algorithm. The TF-IDF is used to determine the importance of a word in a document within a collection or corpus (the corpus in our system is the training set). The importance increases proportionally to the number of times a term appears in a document but is offset by the frequency of the term in the corpus. The TF-IDF is calculated as follows:

$$TF\text{-}IDF\ (t_{ij}) = TF(t_{ij}) * IDF_i \qquad (1)$$

where $TF(t_{ij})$ is Term Frequency that measures how frequently a term t_i occurs in a document d_j. Since the documents are different in length, it is possible that a term would appear more times in longer documents than shorter ones. Thus, the term frequency is normalized using the document length:

$$\mathrm{TF}\left(t_{ij}\right) = \frac{\text{Number of times term } t_i \text{ appears in a document } d_j}{\text{Total number of terms in a document } d_j} \qquad (2)$$

The IDF_i is Inverse Document Frequency which measures the importance of a term t_i across all documents in the training set:

$$\mathrm{IDF}(t_i) = \log\left(\frac{\text{Total number of documents in the training set}}{\text{Number of documents with term } t_i \text{ in the training set}}\right) \qquad (3)$$

The TF-IDF weighted terms are calculated between 0 and 1 for each document in the training set. Therefore, all the concepts in the reference ontology are associated with training documents that have TF-IDF weighted terms, which can be used to measure a vector similarity between a concept represented by the document and a paper that we want to classify.

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

B.2. Classification phase

In this phase, papers from the CiteSeerX dataset are classified to create a database of paper profiles for the recommender system to make recommendations from. The cosine similarity method is used to assign an input paper to appropriate concepts in the reference ontology. In our system, the cosine similarity algorithm [13] is applied to classify an input paper to the correct concepts:

$$SW_j = CosinSim\ (d_j, P) = \frac{\sum_{i=1}^{n}(w_{ij} * w_{iP})}{\sqrt{\sum_{i=1}^{n} w_{ij}^2} * \sqrt{\sum_{i=1}^{n} w_{iP}^2}} \quad (4)$$

where d_j is a document that represents a concept c_j in the reference ontology, P is an input paper, w_{ij} is the TF-IDF weight for term t_i in d_j and w_{iP} is the TF-IDF weight for term t_i in P. The cosine similarity is computed between all concepts' documents and paper P. The output from the classification phase is a profile for representing the research papers, composed of a decreasing ordered list of concepts' IDs along with their cosine similarity (c_j, SW_j) for each input paper P in the dataset. The Cosine Similarity (SW_j) is the degree of association between a paper P and a concept c_j. The resulting profile of papers is stored in a database which is used to build the tree of concepts model for the users and the papers.

C. DNTC user profiling phase

The main goal of the user profiling phase is to build the user profile as a dynamic normalized tree of concepts. Building a user profile as a tree of concepts maintains parent-child relationships between the concepts in the ontology. These relationships can be useful while computing the similarity between a user profile and a paper profile. Normalizing the user's tree of concepts by the number of papers read by the user provides a more accurate comparison between a paper profile and the user profile (which generally involves more than one paper).

All papers read by a user are stored in a user's log file as paper ID associated with a time sequence of the paper's reading order. Hence, the user profile is dynamically updated each time the user reads a new interesting paper (we assume if the user reads a paper, then this is a paper of interest to the user). We added this new feature (time sequence of the paper's reading order) to make the proposed tree profiling model dynamic and changeable because user preferences and interests change over time.

For each paper that is read by the user, the top N related concepts and their corresponding cosine similarity weights are retrieved from the paper's profile, which results from the classification phase. In order to exploit the relationships between concepts in a hierarchical concept ontology, a user tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts. Then, the user tree is updated each time a new paper is read by the user as follows. For every new paper, the top N concepts and their corresponding Cosine Similarity (*SW)* weights are used to update the existing user tree. First, the *SW* weights for the top N concepts are updated by adding the new *SW* weights to old weights values in the user tree. Then, new weight values recursively propagate to the parent nodes until the root node is reached. We assign weights to parents according to the following equation:

$$SW_{\text{Parent}} = \propto \times\ SW_{\text{Child}} \quad (5)$$

Where SW_{Parent} is the weight of the parent, SW_{Child} is the weight of the child and α is the weight propagation factor. α is

```
Build Dynamic Normalized User Tree (UserID, UserTree, PapersProfiles, CurrentTime, Alpha, TopN)
{
    CurrentNumberOfUserPapers =0;
    Foreach Paper Pi in user's log file in CurrentTime do
        {
           CurrentNumberOfUserPapers = CurrentNumberOfUserPapers + 1;
           Get the TopN concepts and their corresponding weights from Paper Pi Profile;
           Foreach concept cj in the TopN concepts do
             {
                 Find the concept cj in the UserTree;
                 Update the concept cj weight: SWj += Pi_SWj;
                 If the concept cj is not root do
                 {
                        currentConcpet = cj ;
                        CurrentConcept_SW = SWj;
                        Loop until UserTree's root reached
                        {
                          Get currentConcpet.Parent;
                          Update currentConcpet.parent weight: SWP += CurrentConcept_SW * Alpha;
                          currentConcpet = currentConcpet.Parent;
                          CurrentConcept_SW = SWP;
                        }
                 }
           }
        }

  //Divide all the concepts' weights by the current total number of user's reading papers.
  Foreach concept cj in UserTree do
    {
        Divide the concept cj weight: SWj = SWj / CurrentNumberOfUserPapers;
    }
}
```

Fig. 4. DNTC user profiling algorithm.

used to maintain the parent-child relationships between the concepts in the user's tree and its value varies between 0 and 1. If α is given the value zero, then the parents will not be assigned any part of the child's weight and there will be no actual tree structure in the user profile, which means a user profile is created as a vector of concepts without any parent-child relationships in a tree structure. Otherwise, if α is given non zero value ($0 < \alpha < 1$), then a user profile will be created as tree of concepts. α is used to determine how much of a child's weight is propagated to its parent. The value of α will be discussed in section IV.B.

Finally, all concept weights are divided by the total number of papers that are read by the user in order to normalize the concept weights. Without normalizing the user's tree of concepts, the concept weights are too large in comparison to the weights in a tree of concepts for a single paper in the recommendation phase. Fig. 4 presents our DNTC user profiling algorithm. The output of the DNTC user profiling phase is a normalized tree of concepts and their corresponding weights. This tree contains all the concepts in the reference ontology. It implicitly encodes a subtree of the sort described in section II above, by eliminating concept nodes with zero weight. However, retaining these nodes in the tree simplifies the Tree Edit Distance algorithm we use below. This dynamic normalized tree is used in the recommendation phase in next section.

D. Dynamic recommendation phase

In this phase, the trees of concepts for all papers that the user has not read (unread papers) are created. Then, the user profile, represented as a dynamic tree of concepts, is compared with the unread papers' trees of concepts to recommend the most relevant papers to the user's interests. The details are as follows.

The outputs from the papers classification phase and the DNTC user profiling phase are used as inputs to this phase. These inputs are: the papers' profiles and the user's DNTC profile. First, a tree of concepts is built for each unread paper in our dataset collection. A tree of concepts for an unread paper is built based on the top *N* concepts and their weights from the paper's profile, stored in the database which resulted from the papers classification phase. The process for building the tree of concepts for a paper is as described above: a tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts, the top N concepts and weights for this paper are retrieved from the profile database, and the weight values are propagated recursively to the parent nodes according to the equation (5). Once the user profile and the papers profiles are represented as trees of concepts, Tree Edit Distance [9] is used to calculate the distance between two trees (the user's tree and a tree of concepts for an unread paper). This distance is the cost of transforming one tree into another with the minimum number of operations. There are three types of operation: insertion, deletion and substitution. Insertion operation is the cost of inserting a new concept into the tree with a given weight. Deletion operation is the cost of deleting an existing concept with a given weight from the tree. Substitution operation is the cost of changing a concept's weight to another weight.

In our 2012 ACM CCS trees we suppose that the concept with zero weight is non existing node. Hence, the cost of deletion or insertion of a concept is equal to the weight associated with the concept. Whereas the substitution cost is the difference between weights of an existing concept in both trees. Thus, the cost of modifying a tree of concepts for a paper to match the user tree is calculated. The two most similar trees are those which have the lower total cost of transformations between them. After calculating the total cost between all trees of concepts for the papers and a user tree, the total cost with its associated id of the paper (PaperID) are stored a list and sorted in increasing order. Hence, the closest papers to user's preferences appear first and the most distant papers appear last. The final output of the recommendation phase is a list of ordered recommended papers. In our system the Tree Edit Distance technique runs dynamically every time the user reads a new paper from the system dataset collection. Fig. 5 presents the algorithm for our Dynamic Tree Edit Distance technique.

```
Dynamic Tree Edit Distance (UserTree, UnreadPapersTrees, CurrentTime)
{
    //m is the number of unread papers in CurrentTime.
    Create an array (ECosts [m]) to save the edit distance costs for each paper;
    Foreach UnreadPaperTree PTi do
        {
           W1=0, W2=0;
           Foreach concept cj in UserTree do
           {
               Get the concept cj weight in UserTree SWUj;
               Find the concept cj in UnreadPaperTree PTi and its weight SWPTij;
               W1 = SWUj;
              W2 = SWPTij;
              Absolute = |W1-W2|;
              Ecost [PTi] += Absolute;
           }
         }

    Sort the array Ecosts [m] in increasing order;
  }
```

Fig. 5. Dynamic Tree Edit Distance technique algorithm.

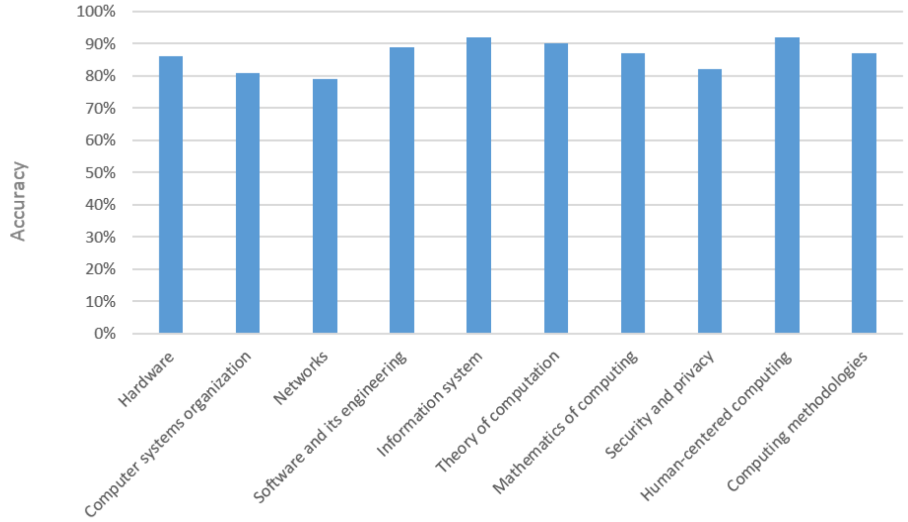

Fig. 6. Accuracy for papers classification phase.

IV. EVALUATION

In order to measure the performance of the proposed system, we evaluate:

1) The accuracy of the classifier model.
2) The performance of our DNTC user profiling and recommendation method.

For these purposes we introduce two evaluation experiments. The first experiment aims to evaluate the classification performance for mapping papers in a dataset. The second experiment evaluates the performance of our DNTC recommendation method. We conduct our evaluations using ACM and CiteSeerX datasets. ACM dataset contains 16,307 mapped papers for 2012 ACM CCS ontology, and is used as the training set. CiteSeerX is a search engine and digital repository of scientific and research papers. It is a collection of over 5 million papers primarily in the field of computer and information science. We used 100,000 papers as a subset of that collection. This subset of CiteSeerX's papers are entered to our classifier to classify them according to 2012 ACM CCS ontology and we then use them as our dataset to evaluate the performance of our DNTC recommender system.

A. *Evaluation of the classification phase*

ACM provided us with a dataset that contains mapped papers for 2012 ACM CCS ontology. The main categories in 2012 ACM CCS that are evaluated are: Hardware, Computer systems organization, Networks, Software and its engineering, Information system, Theory of computation, Mathematics of computing, Security and privacy, Human-centered computing, Computing methodologies. The total number of the concepts under these categories is 1,329 concepts, and the number of leaf concepts is 986 concepts. The papers are mapped by the authors of the papers. The authors of the papers are allowed to assign their papers to more than one leaf concept. To evaluate the accuracy of our classifier, 50% of ACM dataset is used as training set and the other 50% as the test set. The papers from the training set used to learn a concept (c_j) are all combined into one training document file (d_j). All terms in this file are converted to vectors with their weights using the TF-IDF as explained in section III.B.1.

Following this training phase, papers in the test set are classified as explained in section III.B.2. The output for each paper is stored as the paper's profile. If the highest weighted concept resulted from the classifier is one of the concepts that are chosen by the paper's authors, then we consider it as positively classified. We evaluate the performance using the following equation:

$$\text{Accuracy} = \frac{\text{Positive classified papers}}{\text{All papers}} \qquad (6)$$

Fig. 6 shows the accuracy results for our classifier for the main categories in ACM CCS 2012. The accuracy results may depend on distribution of concepts in the training set. For example, the concepts with significant high accuracy result (92%) under Information systems and Human-centered computing may have good representation among the training papers. Whereas the concepts with low accuracy results, such as (79%) under Networks, may have poor representation among the training papers. The average of the classification results in accuracy for all categories is 87%.

After evaluating the accuracy of our classifier, we retrained the classifier using all papers in ACM dataset as training set. We used this classifier to classify the CiteSeerX papers to create the paper profile database which serves as our dataset in all the subsequent experiments.

B. *Evaluating the performance of the DNTC recommender system*

The evaluation process of recommender system algorithms is known to be difficult and expensive as these systems are typically complex and have many components, properties and parameters which have to be examined in order to provide the optimum performance [14]. To establish a preliminary indication of performance, offline evaluations are attractive because they require no interaction with real users, and thus the measuring of performance is allowed at a low cost in terms of time and effort [15]. Therefore, offline evaluation methodology is used for our evaluation to measure the performance of the proposed DNTC recommender system. We opted to use a user behaviour simulation approach to test specific scenarios for multiple concepts and variant range of papers quantity. We have to create user scenarios that simulate users' interests and preferences. We created 9 main templates for user scenarios to simulate different numbers of concepts that represent multiple user interests. We have 3 main types of template scenarios that consider different number of concepts during user's reading: three concepts, four concepts and five concepts as follow:

- Users' template scenarios 1, 2 and 3 consider three concepts (as shown below):

Scenario 1: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	4	6-9
Concept 3	6	10-15

Scenario 2: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	10	1-10
Concept 2	11	11-21
Concept 3	9	22-30

Scenario 3: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	15	1-15
Concept 2	18	16-33
Concept 3	17	34-50

- Users' template scenarios 4, 5 and 6 consider four concepts (as shown below):

Scenario 4: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	5	11-15

Scenario 5: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	6	1-6
Concept 2	9	7-15
Concept 3	7	16-22
Concept 4	8	23-30

Scenario 6: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	14	1-14
Concept 2	11	15-25
Concept 3	13	26-38
Concept 4	12	39-50

- Users' template scenarios 7, 8, and 9 consider five concepts (as shown below):

Scenario 7: small quantity of papers (15)

Concepts	Number of papers	Time sequence for user reading
Concept 1	3	1-3
Concept 2	4	4-7
Concept 3	3	8-10
Concept 4	2	11-12
Concept 5	3	13-15

Scenario 8: medium quantity of papers (30)

Concepts	Number of papers	Time sequence for user reading
Concept 1	5	1-5
Concept 2	7	6-12
Concept 3	6	13-18
Concept 4	4	19-22
Concept 5	8	23-30

Scenario 9: large quantity of papers (50)

Concepts	Number of papers	Time sequence for user reading
Concept 1	9	1-9
Concept 2	11	10-20
Concept 3	10	21-30
Concept 4	12	31-42
Concept 5	8	43-50

Each type has three different scenarios that involve different quantity of papers during user's reading. There are small quantity (15 papers – scenarios 1, 4 and 7), medium quantity (30 papers – scenarios 2, 5 and 8) and large quantity (50 papers, scenarios 3, 6 and 9).

Each scenario template is applied on the ten main categories in ACM CCS 2012: Hardware, Computer systems organization, Networks, Software and its engineering, Information system, Theory of computation, Mathematics of computing, Security and privacy, Human-centered computing, Computing methodologies. Hence, we have 90 virtual users (i.e. 9 templates*10 main categories). The concepts are selected randomly from the main categories in ACM CCS 2012 to create each individual user's scenarios using the templates. The papers for each concept are chosen randomly from our classified CiteseerX dataset that resulted from the classification phase.

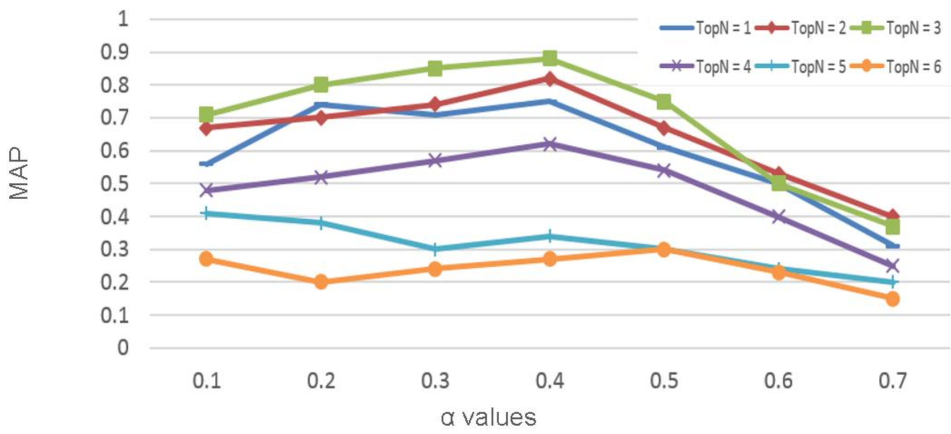

Fig. 7.The MAP results using different α and *TopN* values.

1) *Evaluating α and TopN parameters*

In this section we evaluated different values of α (the propagation factor) and *TopN* (the number of the top related concepts for a paper) parameters to find the optimal values that provide the best overall performance for our recommender system. The measurement that is used for evaluation is Mean Average Precision (MAP). The MAP for all users is the average of the average precision of each user [11]:

$$MAP = \frac{\sum_{i=1}^{M} AVG\ P}{M} \quad (7)$$

We calculated the average precision (AVG P) for each user as follows [12]:

$$\text{AVG P} = \frac{P_{10} + P_{20} + P_{30}}{3} \quad (8)$$

Where P_{10}, P_{20} and P_{30} are precisions for cut-off results for top 10, 20 and 30 recommended papers. The precision for cut-off results at position *k* (P_k) is used to evaluate the top k recommended papers as follows [12]:

$$\mathbf{P_k} = \frac{\textbf{Number of relevant recommended papers to a user}}{\mathbf{k}} \quad (9)$$

Fig.7 shows the MAP results of applying our recommender system on all the users' scenarios using different α and *TopN* values. It can be clearly seen that the MAP results for *TopN*= 6 are relatively low. This is because the top 6 related concepts are a very large number of concepts to be included during build user and paper trees of concepts. The MAP results increase whenever the *TopN* value decreases until *TopN*=3. When *TopN*=3, we have the best results because the top 3 similar concepts to a paper might hold the most essential concepts that are expected to be related to this paper, while considering just the top 1 or 2 concepts may omit some of very significant concepts.

We tested our system with different values for α in the range of [0.1 to 0.7]. Fig.7 shows that the MAP results improve when α value comes close to 0.4 and *TopN* values decrease, and clearly MAP results tend to decrease when reach the smallest or largest values (i.e. 0.1 and 0.7 respectively). The results are very low when α = 0.7, because the propagation value is very large, and then large values are propagated over the reference ontology that makes recommending the correct interests is difficult. When α = 0.1, most of the papers were mapped to leaf concepts from the reference ontology which make the recommendations to be too specific to represent all the users' interests. When α=0.4, the MAP results improve considerably as this value maintains a balance between general and specific concepts. According to these results, we assign α to be 0.4 and *TopN* to be 3 in our system.

2) *Comparing our system against baselines*

In this section we compared our DNTC system against two baselines. Baseline 1 is recommender system using dynamic vector of concepts (DVC) where there is no propagation of weights to parents (i.e. α=0). Baseline 2 is recommender system using non-normalized tree of concepts (NNT) [3].

Fig.8 shows the AVG P for our DNTC system against the two baselines. For user scenarios 1, 2 and 3 that consider only three concepts, we can see that vector of concepts system results are comparable with our DNTC model. However, when the number of concepts are increased in the other scenarios to be more than three concepts, our DNTC system outperforms the DVC method. This is because with multiple concepts the task of user profiling and recommendation is more difficult for the recommender system based on vectors of concepts. For instance, scenarios 7, 8, and 9 consider five concepts during users' reading and there is a substantial improvement in the average precision for these scenarios by using our system. Therefore, when a user reads multiple concepts, our system based on tree of concepts significantly outperforms the system that based on vectors of concepts.

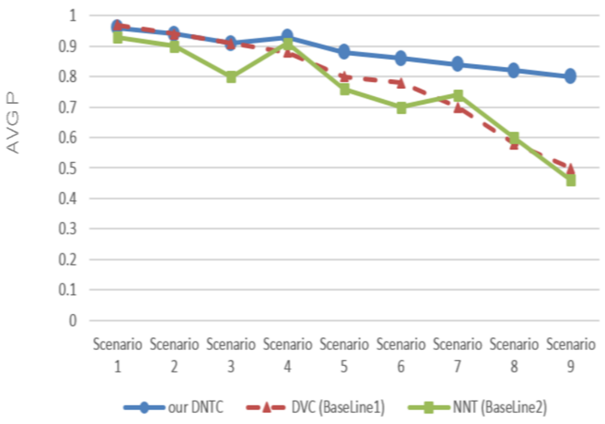

Fig. 8. Comparing average precision for each scenario with the three recommender systems.

With baseline 2 NNT the results in Fig.8 shows that when the quantity of papers is small as in scenarios 1, 4 and 7 that involve 15 papers, the results for NNT system are slightly lower than our system. However, when the quantity of papers is increased to be 30 papers with more than three concepts in scenarios 5 and 8, the results for NNT decline significantly compared with our system. NNT results dramatically drop when the number of papers becomes 50 papers in scenarios 3, 6 and 9. This is because the NNT system does not normalize the concepts' weights in the user's tree of concepts to be appropriate to compare them with the concepts' weights in a paper's tree of concepts. Hence as the user reads more papers, the weights in the user profile grow and become less and less comparable with the weights in the profile of a single paper.

Finally, both the DVC and NNT system achieved the lowest average precision performance in Fig. 8 at scenario 9, where the scenario consider five concepts of interests and 50 papers. The average precision at scenario 9 for DVC is 0.5 and for NNT is 0.46. Whereas the average precision for our DNTC system is 0.8, DNTC system did not drop dramatically as DVC and NNT systems. Therefore, when a user reads multiple concepts and large quantity of papers, our system significantly outperforms both of the baselines systems. Table 1 shows the MAP that reflect the results of those of Fig.8. Our DNTC system has the highest MAP of 0.88, while DVC scored the second best MAP (i.e. 0.78) while NNT achieved the lowest MAP of 0.76.

TABLE I. MAP RESULTS FOR THE THREE SYSTEMS.

Recommender system	MAP
Our system (DNTC)	0.88
Baseline 1 (DVC)	0.78
Baseline 2 (NNT)	0.76

Overall, it can be clearly seen that the proposed DNTC system effectively outperforms the other two systems, and is able to provide high average precision when a user has multiple concepts and read large quantity of papers.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel recommender system for research papers which used a Dynamic Normalized Tree of Concepts (DNTC) user modelling technique. Our system utilizes the ontology for 2012 ACM CCS, which is far richer and more complex than the previous 1998 ACM CCS ontology. The user profiling phase creates a user profile as a dynamic normalized tree of concepts which is used with a dynamic tree edit distance method to compare between the user profile and the new unseen research papers that are also represented as tree of concepts. We performed offline evaluations to find the optimal values for α and *TopN* parameters that can produce the best overall performance for our system. According to our evaluative results, the optimal values are $\alpha = 0.4$ and $TopN = 3$. As part of evaluations we compared our DNTC model with two baselines: recommender system using dynamic vector of concepts (DVC) and recommender system using non-normalized tree of concepts (NNT). Our results show that our novel DNTC model significantly outperforms both DVC and NNT systems when simulating user's reading behavior of large quantity of papers and of multiple topics (concepts). In our future work, we will improve our system to be able to determine multiple concepts reflecting user's long-term and short-term interests.

REFERENCES

[1] G D. H. Park, H. K. Kim, I. Y. Choi, and J. K. Kim, "A literature review and classification of recommender systems research," Expert Systems with Applications, vol. 39, no. 11, pp. 10059-10072, 2012.

[2] "The 2012 ACM Computing Classification System — Association for Computing Machinery," https://www.acm.org/about/class/2012 .

[3] K. Chandrasekaran, S. Gauch, P. Lakkaraju, and H. P. Luong, "Concept-based document recommendations for citeseer authors," In International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems. Springer Berlin Heidelberg, pp. 83-92, 2008.

[4] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," The adaptive web, In: Brusilovsky P., Kobsa A., Nejdl W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, pp. 54-89, 2007.

[5] J. Lee, K. Lee, and J. G. Kim, "Personalized academic research paper recommendation system," arXiv preprint arXiv:1304.5457, 2013.

[6] S. Agarwal, and A. Singhal, "Handling skewed results in news recommendations by focused analysis of semantic user profiles. " 2014 International Conference on Reliability Optimization and Information Technology (ICROIT), pp. 74-79, 2014.

[7] X. Tang and Q. Zeng, "Keyword clustering for user interest profiling refinement within paper recommender systems," Journal of Systems and Software, pp.87-101, 2012.

[8] A. Kodakateri Pudhiyaveetil, S. Gauch, H. Luong, and J. Eno, "Conceptual recommender system for CiteSeerX," In Proceedings of the third ACM conference on Recommender systems, pp. 241-244, 2009.

[9] P. Lakkaraju, S. Gauch, and M. Speretta, "Document similarity based on concept tree distance," In Proceedings of the nineteenth ACM conference on Hypertext and hypermedia, pp. 127-132, 2008.

[10] K. Sparck Jones, and P. Willett, Readings in Information Retrieval. Morgan Kaufmann, San Mateo, US. 1997.

[11] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval." Cambridge University Press, 2008.

[12] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, "Recommender systems: an introduction," Cambridge University Press, 2010.

[13] B. Ricardo and R. Berthier, "Modern Information Retrieval the concepts and technology behind search second edition," Addision Wesley, 2011.

[14] L. Li, L. Zheng, F. Yang, and T. Li, "Modeling and broadening temporal user interest in personalized news recommendation," Expert Systems with Applications, Elsevier, pp. 3168-3177, 2014.

[15] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in Recommender systems handbook, Springer, pp.257-297, 2011.

[16] G. Salton and M. McGill, "Introduction to modern information retrieval,", McGraw-Hill, 1983.

[17] M. Zeb and M. Fasli, "Adaptive user profiling for deviating user interests," IEEE 3rd Conference in Computer Science and Electronic Engineering (CEEC), pp. 65-70, 2011.

[18] "CiteseerX, scientific literature digital library and search engine," http://citeseerx.ist.psu.edu/index.

A Novel Short-term and Long-term User Modelling Technique for a Research Paper Recommender System

Modhi Al Alshaikh , Gulden Uchyigit and Roger Evans
School of Computing, Engineering and Mathematics, University of Brighton, Brighton, United Kingdom
{m.alalshaikh, g.uchyigit, r.p.evans}@brighton.ac.uk

Keywords: Recommender system, Personalization, User profile, Research papers, Short-term, Long-term.

Abstract: Modelling users' interests accurately is an important aspect of recommender systems. However, this is a challenge as users' behaviour can vary in different domains. For example, users' reading behaviour of research papers follows a different pattern to users' reading of online news articles. In the case of research papers, our analysis of users' reading behaviour shows that there are breaks in reading whereas the reading of news articles is assumed to be more continuous. In this paper, we present a novel user modelling method for representing short-term and long-term user's interests in recommending research papers. The short-term interests are modelled using a personalised dynamic sliding window which is able to adapt its size according to the ratio of concepts per paper read by the user rather than purely time-based methods. Our long-term model is based on selecting papers that represent user's longer term interests to build his/her profile. Existing methods for modelling user's short-term and long-term interests do not adequately take into consideration erratic reading behaviours over time that are exhibited in the research paper domain. We conducted evaluations of our short-term and long-term models and compared them with the performance of three existing methods. The evaluation results show that our models significantly outperform the existing short-term and long-term methods.

1 INTRODUCTION

A major challenge in recommender systems is the modelling of dynamically evolving short-term and long-term user's interests. The short-term interests represent the user's most recent interests which are more erratic, whereas the long-term interests are more stable in comparison (Challam et al., 2007). Recommender systems for research papers suffer from a number of limitations; for example, fast deviations in short-term interests may remain undetected and stable long-term interests may not be appropriately updated to reflect the user's evolving short-term and long-term interests. The importance of this stems from the need to design automatically adaptable user profiling techniques that should keep track of multiple information that is needed by the user. It is important to recommend right papers at the right time. Therefore, there is a need for user profiling models and techniques that automatically adapt to the diverse and frequently changing users' short-term and long-term interests.

Existing short-term and long-term user modelling techniques have been developed for domains such as recommending web pages (Gao et al., 2013; Hawalah and Fasli, 2015; Li et al., 2007) and news articles (Zeb and Fasli, 2011; Agarwal and Singhal, 2014; Zeb and Fasli, 2012), where a user reading behaviour is different from the research paper domain. These models depend on continuous time-based user behaviour measured in days for the web pages domain and in hours in the news domain. These models also assume that users are continuously active in their reading with no significant breaks.

In this paper, we present analysis of users' reading behaviour of research papers using the BibSonomy dataset (Knowledge & Data Engineering Group, 2017). The BibSonomy dataset contains actual records of users' interests as posts for research papers. We consider these posts as users' reading records of research papers. Our analysis shows that users are actively reading during some days and inactive on other days. Moreover, they may also be inactive for several months. Furthermore, the users have different reading behaviours from each other, and reading behaviour for a user may change during a year. Therefore, utilizing continuous time-based models for building a user's profile based on continuous timing algorithms (such as Hawalah and Fasli, 2015) or time-based window (such as Gao et al., 2013) are not appropriate. In this paper, we propose a novel user modelling method for short-term and long-term interests as follows:

a. *Short-term model:* this model is based on a novel personalized dynamic sliding window (PDSW) technique where the window length is adapted according to the ratio between the number of concepts/interests and number of papers recently read by the user. The content of these papers are then used to build the user's short-term profile.
b. *Long-term model:* this model determines the user's long-term concepts/interests and then selects papers that represent those concepts/interests. The user's long-term profile is built from the selected papers.

The rest of this paper is organized as follows. Section 2 analyses users' reading behaviour of research papers using the BibSonomy dataset. Section 3 presents our short-term and long-term models. Section 4 presents evaluation and results produced by our models. Finally, the conclusions are presented in section 5.

2 ANALYZING USERS' READING BEHAVIOUR OF RESEARCH PAPERS USING THE BIBSONOMY DATASET

The BibSonomy dataset contains actual records of users' interests as posts for research papers over approximately a ten-year period. Each post contains: metadata for a research paper, date and time of the post. We consider these posts as users' reading records of research papers. For our analysis, we used records of users' reading behaviour over the last two years 2015 and 2016 for users in computing area. This included analysis of 1,642 user records and 43,140 research papers. Our analysis involved automatically searching for patterns of users' reading behaviour. Firstly, we analysed the periods of days and months that a user was inactive (an inactive day/month is a day/month that the user did not read

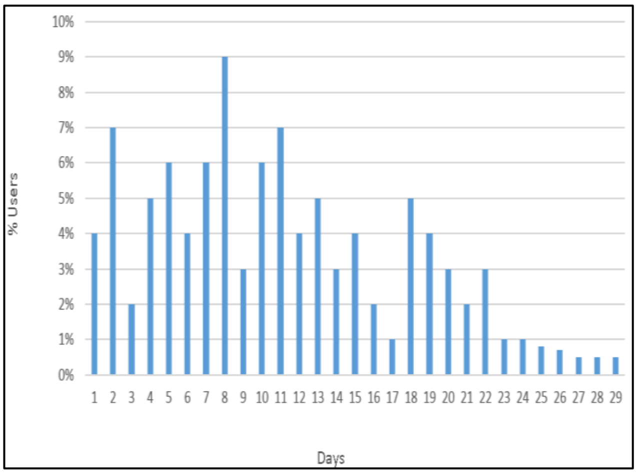

Figure 1. Average inactive days in one active month.

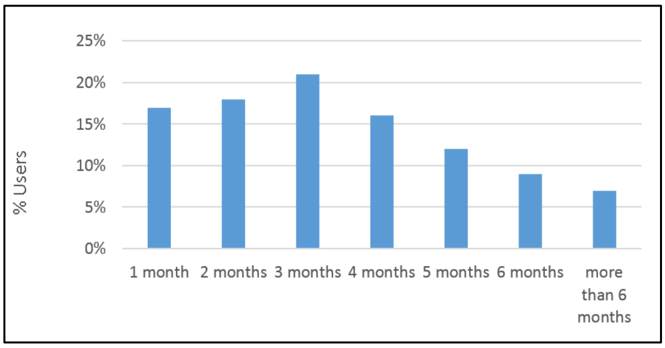

Figure 2. Average inactive months.

any papers). Secondly, we analysed the users' reading behaviour during active months.

We analysed the periods of days and months that a user was inactive as follows:

a. Average number of consecutive inactive days during one active month. (An inactive day is a day that the user did not read any papers.)
b. Average consecutive inactive months. (An inactive month is a month that the user did not read any papers.)

Figure1 shows the average number of consecutive inactive days in one active month. It can be seen that users are not active every day; they do not read papers continuously. Also, users have different patterns of this short-term inactivity. For example, 9% of users are inactive for eight days per active reading month. Therefore, using a fixed duration in time-based models for short-term user profiling is not suitable in this domain. This is because the users can be inactive for several days, which will lead to inaccuracies if modelled based on fixed time periods.

Figure 2 presents the average consecutive inactive months. Our results show that users may not read for several months and may have long inactive periods. For example, our results show that 21% of users are inactive in reading papers for three continuous months.

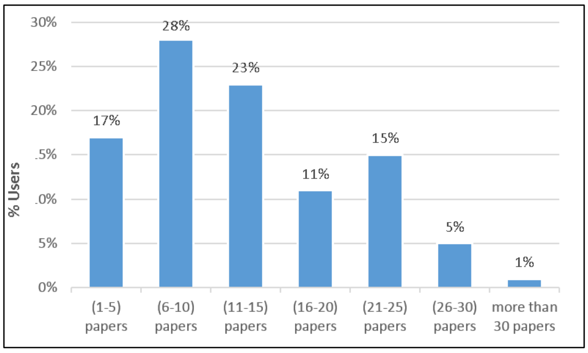

Figure 3. Average number of papers per active month.

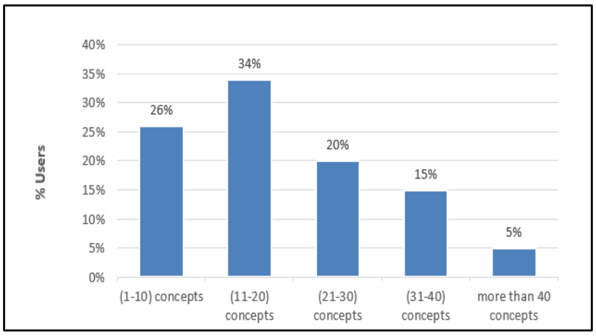

Figure 4: Average number of concepts per active month.

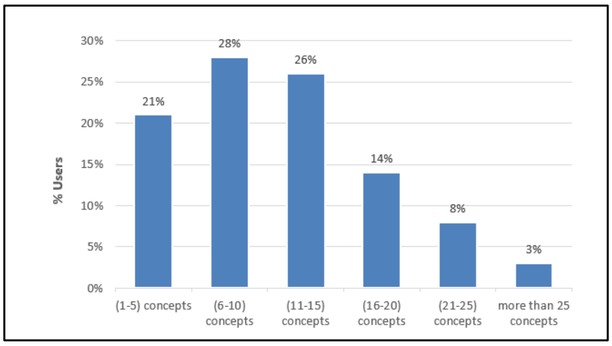

Figure 5: Number of long-term concepts.

Our analysis for the users' behaviour during active months includes the following:

a. Average number of papers that are read by a user per active month.
b. Average number of concepts/interests encountered in a user's reading per active month.
c. Number of long-term concepts that stay in a user's record more than one active month.

Figure 3 shows the average number of papers read by a user per active month. There is significant variability in the number of papers read by users in one active month. For example, 28% of the users read 6-10 papers and 23% of the users read 11-15 papers per one active month.

We analyse average number of concepts per one active month as follows. From the BibSonomy metadata we extracted papers' title, abstract and keywords. Then, each paper is entered to the classifier in our earlier work (Al Alshaikh et al., 2017) to classify it to the three most closely related concepts in 2012 ACM Computing Classification System (CCS) ontology (ACM, 2012).

Figure 4 shows the average number of concepts that are encountered by a user per active month. Figure 5 presents number of long-term concepts that remain in a user's record for more than one active month. It can be seen that the number of long-term concepts in Figure 5 are fewer than the number of concepts in Figure 4. For example, the largest group of users in Figure 4 (34%) encounters 11-20 concepts per month, whereas the largest group of users in Figure 5 (28%) have 6-10 concepts remaining for more than one active month. This is because some of the concepts represented in Figure 4 can be short-term interests. Not all the short-term concepts can be considered as being long-term concepts. The current recommender systems for research papers do not involve short-term and long-term models; they mostly use the whole user reading history. Hence, they are not efficient in recommending the right papers at the right time for evolving users' interests. Therefore, it is important to develop short-term and long-term models for a research paper recommender system. The next section presents our novel short-term and long-term models.

3 SHORT-TERM AND LONG-TERM USER MODELS

In this section, we present our novel short-term and long-term models which automatically adapt to different users' reading behaviour.

3.1 Short-term Model

The short-term model uses novel personalized dynamic sliding window (PDSW) technique. The PDSW length is the number of latest papers that are read by a user. These papers are then used to build a short-term user's profile, represented as Dynamic Normalized Tree of Concepts (DNTC) as in our earlier work (Al Alshaikh et al., 2017). Figure 6 presents the basic idea of our short-term model. In Figure 6 the PDSW length is four papers. P_1 is the first paper read by the user, P_2 is the second paper and so on, the current time is T and the short-term user's DNTC tree is U_T.

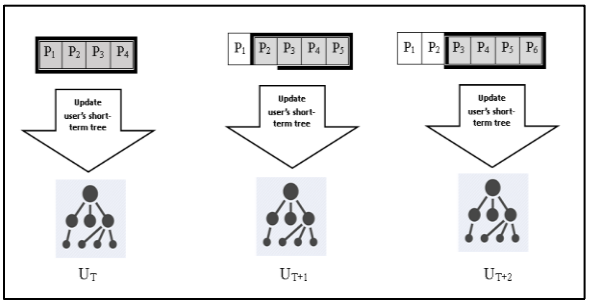

Figure 6: Building DNTC using our short-term dynamic window.

The PDSW length is modified according to the ratio between number of concepts and number of papers that are read by the user. The ratio is calculated for the previous active reading days for a user and results in the length of the sliding window to extend or shrink according to the user's behaviour. The ratio *R* on time *T* is calculated as follows:

$$R_T = \frac{\sum_{i=1}^{PAD_T} \frac{nC_i}{nP_i}}{PAD_T} \qquad (1)$$

where PAD_T is the number of previous active days on time T, *nCi* is the number of concepts in active day *i* and *nPi* is the number of papers in active day *i*. Each time a new paper is read by a user, the new ratio R_{T+1} is compared with the previous ratio R_T. If R_{T+1} is larger than R_T, then the previous PDSW length has a greater distribution of concepts. Hence, we shrink the PDSW length to focus on the latest papers and concepts to discover the new short-term interests. If R_{T+1} is smaller than R_T, then we extend the PDSW length. If $R_{T+1}$is equal to the R_T then the window length remains unchanged. To shrink or extend the length (L) of PDSW, Signum function[1] (sgn) is used as follows:

$$L_{T+1} = L_T + \beta * sgn\,(R_T - R_{T+1}) * R_{T+1} \qquad (2)$$

Where L_{T+1} is the new window length on time *T+1*, L_T is the previous window length on time *T*, β is decay factor and sgn function as follows:

$$sgn\,(R_T - R_{T+1}) = \begin{cases} -1\; if\; R_T - R_{T+1} < 0 \\ 1\; if\; R_T - R_{T+1} > 0 \\ 0\; if\; R_T - R_{T+1} = 0 \end{cases}$$

After calculating the new PDSW length, the latest papers that are read by the user are selected to represent the user's short-term profile. The number of selected papers is an integer equal to the PDSW length. Then, the short-term user's profile is represented as DNTC profile as in (Al Alshaikh et al., 2017). Dynamic Tree Edit Distance technique as in (Al Alshaikh et al., 2017) is then used to recommend a set of papers to the user that match his/her short-term interests.

[1] https://calculus.subwiki.org/wiki/Signum_function

3.2 Long-term Model

The long-term model is updated at the end of each active month for a user. Long-term concepts are the concepts that remain for more than one active month in a user's record. The long-term model selects the papers that represent long-term concepts, then these papers represent a user's long-term profile. The set of long-term concepts is defined as $LC = \{Lc_1, Lc_2,.., Lc_n\}$, where n is the total number of long-term concepts. After selecting the long-term concepts, the papers that are related to at least one of the long-term concepts are selected to represent a user's long-term profile. The set of long-term papers is defined as $LP = \{Lp_1, Lp_2,.., Lp_m\}$, where m is the total number of long-term papers and Lp_i is related at least to one of LC concepts. Then the set of papers LP is used to build a user's long-term DNTC as in (Al Alshaikh et al., 2017). Then, the Dynamic Tree Edit Distance technique (Al Alshaikh et al., 2017) is used to recommend a set of papers to the user that match his/her long-term interests.

4 EVALUATIONS

4.1 Evaluation of short-term model

We evaluated the performance of our short-term model using the BibSonomy dataset. The BibSonomy dataset in section 2 was pruned to remove users with fewer than 60 active days (an active day is a day that the user reads at least one paper). The remaining dataset consists of 1,074 users in the year 2015 and 2016. Every day in the 60 active days for each user is evaluated. The **training set for an active day *i*** is the papers in the user's record for previous active days before the active day *i*. The **testing set for an active day *i*** is the papers that exist in day *i* and the next 29 calendar days in the user's record (we assume that the duration for short-term interests is 30 calendar days). At every active day *i*, if a recommended paper exists in its testing set, then it is relevant to his/her short-term interests. The measurement that is used for evaluation is precision at top *k* papers of an active day *i* for a user *a* as follows:

$$P_k(d_i, a) = \frac{NPi, a}{k} \quad (3)$$

where $NP_{i,a}$ is the number of recommended papers that match the testing set for active day *i* for user *a*. Then, the average precision is calculated for all users *U* for an active day *i* as follows:

$$AVG\ P_i = \frac{\sum_{a=1}^{U} P_k(d_i, a)}{U} \quad (4)$$

The mean average precision for all active days is calculated for all active days (*AD*) as follows:

$$MAP = \frac{\sum_{i=1}^{AD} \text{AVG P}_i}{AD} \quad (5)$$

4.1.1 Evaluating β parameter

In this section we evaluated different values of β (the decay factor in equation 2) parameter to find the optimal value that provide the best overall performance for our short-term model. The optimal value of the decay parameter β was determined by measuring the precision of the model for different values of β. The measurement that is used for evaluation is precision at top 10 papers (*k*=10). Figure 7 presents the MAP for all users using different values of β in the range of [0.1 to 1]. When β = 0.1, the PDSW length is very small to detect the short-term interests. The results increase when the β value increases until β = 0.6, where MAP is 0.76. Then, the PDSW length becomes very large and may include some of the old short-term interests that do not belong anymore to the user's current short-term interests. The value of β used in our model was therefore β = 0.6.

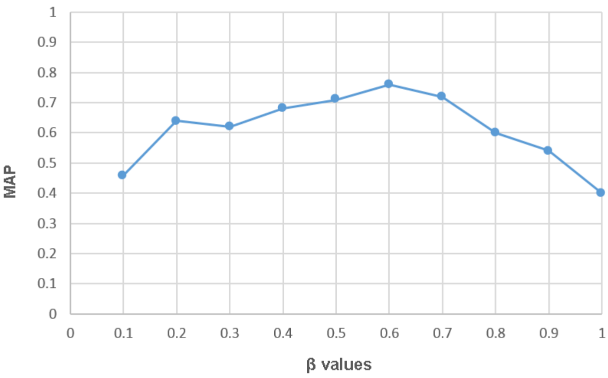

Figure 7. MAP results using different β values for PDSW.

4.1.2 Comparing our short-term model against baselines

We compared our PDSW short-term model against three baselines:

1. DNTC system (Al Alshaikh et al., 2017).
2. Static window time-based model in (Gao et al., 2013).
3. Dynamic time-based model for short-term model in (Hawalah and Fasli, 2015).

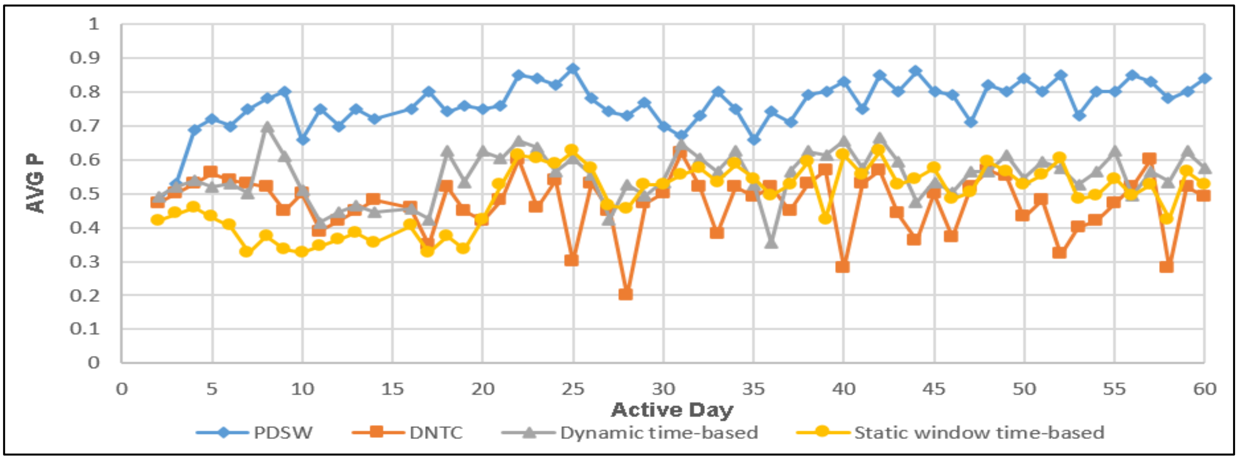

Figure 8. Comparing average precision for our short-term model against baselines.

Our PDSW short-term model and the three systems are run for each day during the 60 active days. Figure 8 shows the overall comparison for our short-term model against the three systems over 60 active days. Table 1 shows the MAP that reflect the results of those of Figure 8. It can be seen that the DNTC system achieves the lowest precision performance with MAP over the 60 active days of 0.47. The DNTC system does not consider short-term behaviour but includes all the papers read by a user. Considering all previous papers in a user's record give the previous existing concepts high weights in a user's profile, hence they are considered as short-term interests. However, new concepts receive lower weights in a user's profile, which can cause sharp drops in the precision in some active days. When it comes to the Static window time-based system, the performance is slightly better than the DNTC system with MAP of 0.49. This is because this system considers only the latest papers during the static window time-based. The low performance of this system because it assumes a user's reading behaviour is static, whereas in reality the user behaviour changes over time. Moreover, each user has different personalized behaviour. When it comes to the Dynamic time-based system, there is improvement in the performance with MAP of 0.55. This system is better than the previous two systems because it can handle the situation when new short-term concepts arise in a user's profile, and it does not depend on static time-based behaviour. However, it has a limitation that it cannot handle the problem of different inactive days for different users' behaviour. Our PDSW system achieves MAP of 0.76 which is an improvement on each of the previous three systems. These results show that our short-term model can effectively learn different users' reading behaviour even if there are different patterns of inactive days. Moreover, it dynamically adapts with the changes in a user's reading behaviour over time.

Table 1: MAP results for the four short-term systems.

System	MAP
DNTC	0.47
Static window time-based	0.49
Dynamic time-based	0.55
PDSW	0.76

4.2 Evaluation of the Long-term Model

We evaluated the performance of our long-term model using the BibSonomy dataset. The BibSonomy dataset in section 2 was pruned to remove users with fewer than 12 active months during the years 2015 and 2016 (an active month is a month that the user reads at least one paper). The remaining dataset consists of 261 users. Every month in the 12 active month for each user is evaluated. The **training set for an active month *i*** is the papers in the user's record for previous active months before the month *i*. The **testing set for an active month *i*** is the papers that exist in in the rest of the user's record and one of its concepts is long-term concept 'LC'. At every active month *i*, if a recommended paper exists in its testing set, then it is relevant to his/her long-term interests. The measurement that is used for evaluation is precision at top *k* papers of an active month *i* for a user *a* as follows:

$$P_k(m_i, a) = \frac{MPi, a}{k} \qquad (6)$$

Where $MP_{i,a}$ is the number of recommended papers that are exist in the testing set for active month *i* for user *a*.

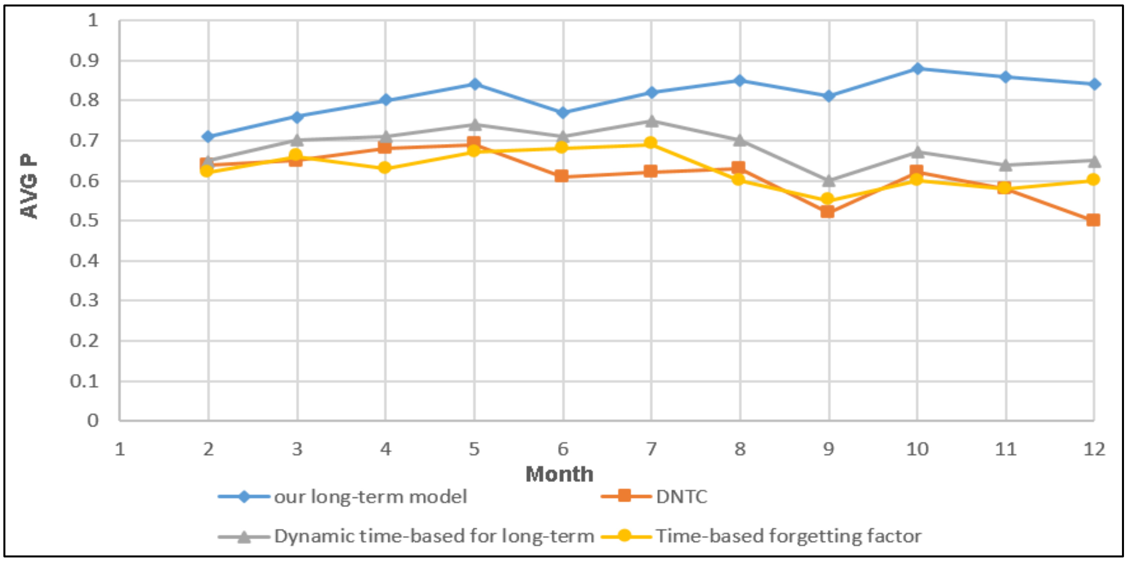

Figure 9. Comparing average precision for our long-term model against baselines.

Then, average precision is calculated for all users *U* for active month *i* as follows:

$$AVG\ P_i = \frac{\sum_{a=1}^{U} P_k(m_i, a)}{U} \quad (7)$$

The mean average precision for all active months is calculate for all active months (*AM*) as follows:

$$MAP = \frac{\sum_{i=1}^{AM} \text{AVG P}_\text{i}}{AM} \quad (8)$$

We compared our long-term model against three baselines:

1. DNTC system (Al Alshaikh et al., 2017).
2. Time-based forgetting factor model in (Gao et al., 2013).
3. Dynamic time-based for long-term interests in (Hawalah and Fasli, 2015).

Our long-term model and the three systems are run at the end of each active month for each user. The top 10 recommended papers (k=10) are evaluated. Figure 9 shows the overall comparison for our long-term model against the three systems over 12 months. Table 2 shows the MAP that reflect the results of those of Figure 9. It can be seen from Figure 9 and table 2 that the DNTC achieves the lower precision performance with MAP of 0.61. After the fifth month DNTC performance declined dramatically because of cumulative calculations for all the papers that are read by the user. This low performance is because DNTC includes all the papers in a user's record even the papers for short-term interests. When it comes to the time-based forgetting factor model, the performance is slightly better than the DNTC with MAP of 0.63. This is because this model has a forgetting factor. However, this forgetting factor is fixed for all users and does not consider different users' behaviour. When it comes to the Dynamic time-based model for long-term interests, there is improvement in the performance with MAP of 0.68. This model is better than the previous two models because it can handle the situation when there is short-term concepts and long-term concepts, and it does not depend on static time-based technique. However, it has a limitation that it does not handle well long inactive periods in users' behaviour. Therefore, after the seventh month its performance declined significantly. Our long-term model achieves MAP of 0.81 which is better than each of the previous three models. This is because our model can effectively learn different users' reading behaviour even if there are different long inactive periods. Our long-term model significantly outperforms the other three baselines after the seventh month as shown in Figure 9.

Table 2: MAP results for the four long-term systems.

System	MAP
DNTC	0.61
Time-based forgetting factor	0.63
Dynamic time-based	0.68
Our long-term model	0.81

5 CONCLUSIONS

In this paper, we presented our novel short-term and long-term models for a research paper recommender system. First, we analysed users' reading behaviour in the BibSonomy dataset. Our analysis shows that the users' reading of research papers is different to that of reading web pages and news articles. Therefore, we developed our short-term and long-term models based on our analysis of users' reading behaviour for the research paper domain. Our evaluations of performance demonstrate that our models significantly outperforms the other baseline systems. Our short-term PDSW model achieves MAP of 0.76 and our long-term model achieves MAP of 0.81. The performance advantage is because our models can effectively learn different users' reading behaviour. Moreover, they dynamically adapt to the changes in users' reading behaviour over time. In future work, we will combined our short-term and long-term models and add collaborative model to develop a hybrid system for the research paper domain.

REFERENCES

ACM Computing Classification System, 2012, URL: https://www.acm.org/about/class/2012.

Agarwal, N., Haque, E., Liu, H. and Parsons, L., 2005. Research paper recommender systems: A subspace clustering approach. *In Advances in Web-Age Information Management*, Springer, pp.475-491.

Agarwal, S. and Singhal, A., 2014. Handling skewed results in news recommendations by focused analysis of semantic user profiles. *In IEEE International Conference on Optimization, Reliabilty, and Information Technology (IEEE ICROIT)*, pp. 74-79.

Al Alshaikh, M., Uchyigit G. and Evans, R, 2017. A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling. *In IEEE Eleventh International Conference on Research Challenges in Information Science (IEEE RCIS 2017)*, pp.200-210.

Challam, V., Gauch, S. and Chandramouli, A., 2007. Contextual search using ontology-based user profiles. *In Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*, pp. 612-617.

Gao, Q., Xi, S.M. and Im Cho, Y., 2013. A multi-agent personalized ontology profile based user preference profile construction method. *In IEEE 44th International Symposium on Robotics (ISR)*, pp. 1-4.

Hawalah, A. and Fasli, M., 2015. Dynamic user profiles for web personalisation. *Expert Systems with Applications, 42(5)*, pp.2547-2569.

Knowledge & Data Engineering Group, University of Kassel: Benchmark Folksonomy Data from BibSonomy, version of January 1st, 2017.

Li, L., Yang, Z., Wang, B. and Kitsuregawa, M., 2007. Dynamic adaptation strategies for long-term and short-term user profile to personalize search. *In Advances in Data and Web Management*. Springer. pp. 228-240.

Zeb, M.A. and Fasli, M., 2011. Adaptive user profiling for deviating user interests. *In Computer Science and Electronic Engineering IEEE Conference (CEEC)*, pp. 65-70.

Zeb, M.A. and Fasli, M., 2012. Dynamically Adaptive User Profiling for Personalized Recommendations. *In IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pp. 604-611.

Predicting Future Interests in a Research Paper Recommender System Using a Community Centric Tree of Concepts Model

Modhi Al Alshaikh , Gulden Uchyigit and Roger Evans
School of Computing, Engineering and Mathematics, University of Brighton, Brighton, United Kingdom
{m.alalshaikh, g.uchyigit, r.p.evans}@brighton.ac.uk

Keywords: Recommender systems, Collaborative filtering, Information retrieval, Research paper recommendations.

Abstract: Our goal in this paper is to predict a user's future interests in the research paper domain. Content-based recommender systems can recommend a set of papers that relate to a user's current interests. However, they may not be able to predict a user's future interests. Collaborative filtering approaches may predict a user's future interests for movies, music or e-commerce domains. However, existing collaborative filtering approaches are not appropriate for the research paper domain, because they depend on large numbers of user ratings which are not available in the research paper domain. In this paper, we present a novel collaborative filtering method that does not depend on user ratings. Our novel method computes the similarity between users according to user profiles which are represented using the dynamic normalized tree of concepts model using the 2012 ACM Computing Classification System (CCS) ontology. Further, a community-centric tree of concepts is generated and used to make recommendations. Offline evaluations are performed using the BibSonomy dataset. Our model is compared with two baselines. The results show that our model significantly outperforms the two baselines and avoids the problem of sparsity.

1 INTRODUCTION

Most research paper recommender systems suggest research papers which are similar to a user's profile which result in a limited set of recommendations based on current user preferences that are represented in the system (Kotkov et al., 2016). A major challenge in recommender systems is to explore the potential of future interests of users (Yang et al., 2016). Content-based approaches are able to recommend a set of papers that relate to user's current interests. However, they suffer from the problem of content overspecialization because they depend only on the metadata of papers in the user's profile; therefore the user is restricted to getting recommendations similar to papers already defined in his/her profile (Isinkaye et al., 2015). Collaborative filtering approaches have the ability to explore potential future interests. Existing collaborative approaches have been developed for domains such as movies, music and e-commerce products. These collaborative approaches are not appropriate for the research paper domain, because they depend on large numbers of user ratings. However, there is a lack of ratings in the research paper domain (Yang et al. 2009). For example, the implicit ratings (users' access logs) on Mendeley[1] (research paper domain) has been compared with Netflix[2] (movie domain), has been found that the sparsity of Mendeley was three orders of magnitude higher than on Netflix (Beel et al., 2016). This is due to the different behaviour of users in these two domains. For example in the movie domain there are many users who have watched the same movies. Therefore, similar users can be found for most users and hence recommendations can be made effectively. However, the research paper domain suffers from the data sparsity problem, where several new papers have not been read by any user and further, a new user may read only a few papers (Jain 2012; Beel et al., 2016). This leads to an inability to successfully locate similar users and hence leads to the generation of weak recommendations.

[1] http://www.mendeley.com/
[2] https://www.netflix.com/gb/

In this paper, we present a new collaborative filtering model that does not depend on users' rating. Our novel method computes the similarity between users according to the users' profiles represented as Dynamic Normalized Tree of Concepts (DNTC) model as in our earlier work (Al Alshaikh et al., 2017). The concepts are the categories in the 2012 ACM CCS ontology (ACM, 2012). The similarity is computed by using the Tree Edit Distance algorithm (Lakkaraju et al., 2008). Then, a Community-Centric Tree of concepts (CCT) is created. The CCT is used to recommend a set of papers that may relate to the user's future interests. We conducted offline evaluations using the BibSonomy dataset (Knowledge & Data Engineering Group, 2017), which contains actual records of users' posts of research papers. Our model is compared with two baselines: content-based DNTC (Al Alshaikh et al., 2017) and User-based Collaborative filtering (UBCF) as in (Nadee et al., 2013). Our model significantly outperforms the two baselines. This is because it maintains the parent-child relationships between the concepts from the 2012 ACM CCS ontology, it considers other potential interests that can be extracted from similar users to the target user, and it avoids the problem of sparsity. The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 presents our model. Section 4 presents evaluations and results. Finally, the conclusions and future work are presented in section 5.

2 RELATED WORK

Most recommender systems in the research paper domain use content-based approaches; for example, the systems that are developed by Chandrasekaran et al. (2008), Kodakateri et al. (2009), Tang and Zeng (2012), and Al Alshaikh et al. (2017). Each of these approaches use ontologies in their user profiling models. Using ontologies provides a significant improvement in the performance of the recommender systems (Gauch et al., 2007). Gauch et al. (2007) noted that most researchers who used ontologies for user profile representation use them in a similar way to weighted keywords where the concepts are represented as vectors of weighted features. Tang and Zeng (2012) and Kodakateri et al. (2009) use vectors of concepts from a predefined ontology to represent user profiles. The ontology that is used in (Tang and Zeng, 2012) is from Sciencepaper Online (Sciencepaper, 2012). Kodakateri et al. (2009) use the '98 ACM CCS ontology (ACM, 1998). The vector of concepts method assumes that the concepts are independent of each other, which is not an accurate representation of the user's preferences (Chandrasekaran et al., 2008). Chandrasekaran et al., (2008) represents the user profile as a tree of concepts. In this technique, the parent-child relationships between the concepts from '98 ACM CCS ontology are maintained whilst computing the similarity between a user profile and the new research papers to be recommended. However, their user profiling model using the tree of concepts technique is static over time, whereas user preferences and needs are not static but change over time. Moreover, this user profiling technique does not normalize the concept weights. Without normalization, the weights in the user's tree of concepts profile representation are too large to compare accurately with the weights in a tree of concepts for a paper in the recommendation phase. To overcome these problems, Al Alshaikh et al. (2017) developed the Dynamic Normalized Tree of Concepts (DNTC) model for user profiles using the 2012 ACM CCS ontology.

Content-based approaches can capture users' current interests, then recommend a set of papers that may related to their current interests. However, content-based approaches are not able to predict users' future interests. Collaborative filtering approaches have the ability to explore potential future interests. There are two major categories of collaborative filtering approaches: the *memory-based* and *model-based* approaches (Shi et al., 2014; Isinkaye et al., 2015). The *memory-based* approaches involve user-based or item-based techniques. In user-based techniques a user-item rating matrix is given, then a user-based technique predicts a user's rating on a target item by combining the ratings that similar users have previously given to that item (Shi et al., 2014). Item-based filtering techniques predict a user's rating using the similarity between items and not the similarity between users. It builds a model of item similarities based on information about other items that a user has previously rated (Deshpande and Karypis, 2004). *Model-based* approaches use the ratings in user-item matrix as input to train prediction models (Ekstrand et al., 2011). These trained prediction models are used to generate recommendations for the users. For example, the matrix factorization model is used in (Gordon et al., 2008) and feedforward neural network model is used in (Vassiliou et al., 2006).

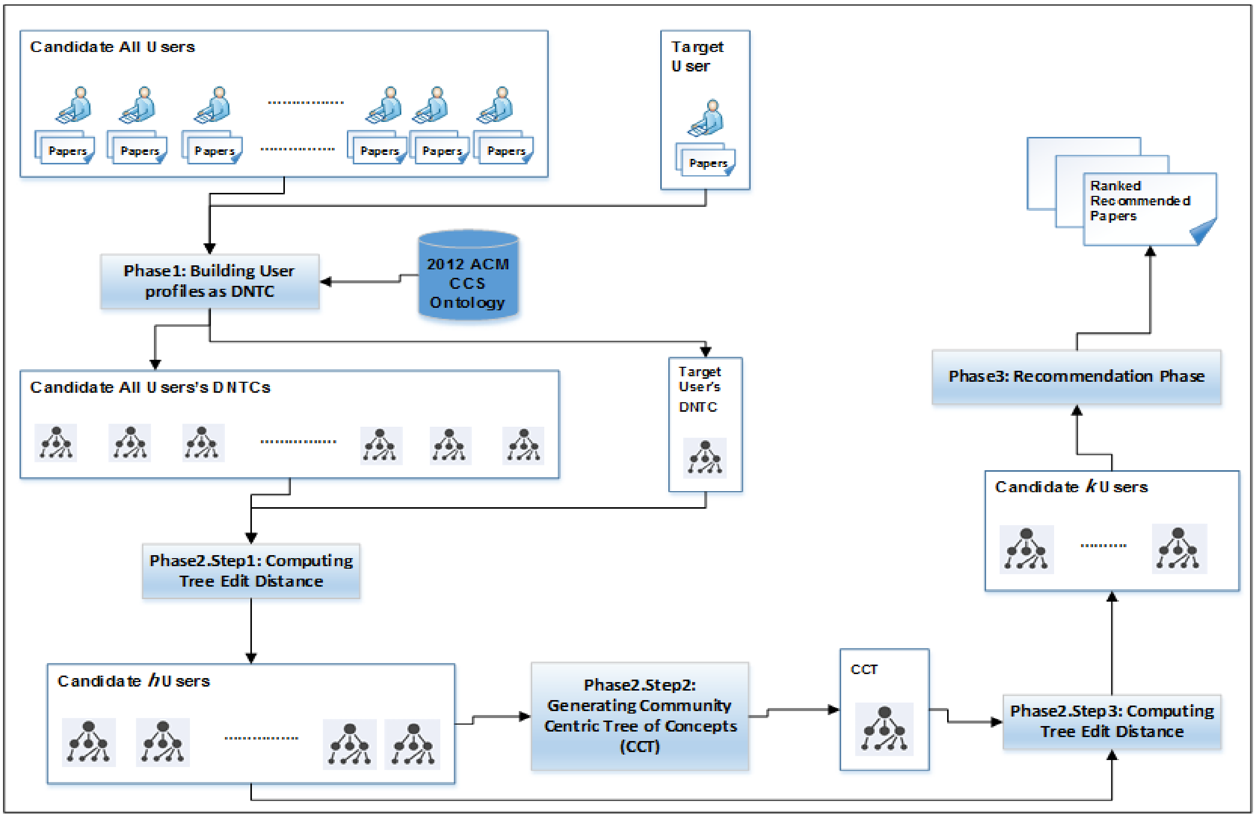

Figure 1: Our collaborative recommendation model.

Nevertheless, the existing collaborative approaches are not appropriate for the research paper domain because they depend on a large number of users' rating, where there is a lack of rating in research paper domain (Yang et al., 2009 and Beel et al., 2016). Nadee et al. (2013) tried to solve the lack of users' rating problem in book recommendation domain. They presented a recommendation approach that considers both the similarity between users and items, and items' popularity to overcome the overspecialization problem. However, their recommendation results are not sufficiently effective for research paper domain. To overcome the problem of lack of users' rating, we have developed a new collaborative filtering model that does not depend on users' rating, which we introduce in the next section.

3 OUR MODEL

The proposed recommendation model is comprised of three phases:

1- Building user profiles as Dynamic Normalized Trees of Concepts using the 2012 ACM CCS ontology.
2- Computing the similarity between the target user and candidate users, then generating a "Community-Centric Tree of concepts" (CCT) for the target user.
3- Recommending a ranked list of research papers for the target user based on CCT.

Figure 1 presents our collaborative recommendation model.

3.1 Phase 1: Building User Profile as DNTC

The main goal of this phase is to build a user profile as Dynamic Normalized Tree of Concepts (DNTC) as in our earlier work (Al Alshaikh et al., 2017). The BibSonomy dataset is used to create a database of users and the papers which they have read. This phase involves two steps: classifying the papers read by the users to the related concepts in the 2012 ACM CCS ontology and building a DNTC profile for each user.

3.1.1 Classifying Papers

The papers that are read by the users are classified to create profiles of the papers for the recommender system. For classification, we used the TF-IDF weighting algorithm and cosine similarity in our classifier (Al Alshaikh et al., 2017). The cosine similarity (SW_j) between a paper and a concept c_j is the degree of association between the paper and the concept c_j. Each paper in the BibSonomy dataset is classified to the three most closely related concepts in the 2012 ACM CCS ontology and stored in the paper's profile along with their cosine similarity. The resulting profile of each paper is stored in the database which is used to build the DNTC profile for each user.

3.1.2 Building DNTC for Each User

Building a user profile as a DNTC maintains parent-child relationships between the concepts from the ontology. These relationships can be useful while computing the similarity between two users' profiles. For each paper that is read by the user, the top three related concepts and their corresponding cosine similarity weights are retrieved from the paper's profile, which results from the classification phase. In order to exploit the relationships between concepts in a hierarchical concept ontology, a user tree of 2012 ACM CCS ontology is initiated with zero weights for all concepts. Then, the user tree is updated each time a new paper is read by the user as follows. For every new paper, the top three concepts and their corresponding cosine similarity weights (*SW*) are used to update the existing user tree. First, the *SW* weights for the top three concepts are updated by adding the new *SW* weights to old weights values in the user tree. Then, new weight values recursively propagate to the parent nodes until the root node is reached. We assign weights to parents according to the following equation:

$$\mathrm{SW}_{\mathrm{Parent}} = \propto \times \ \mathrm{SW}_{\mathrm{Child}} \qquad (1)$$

Where SW_{Parent} is the weight of the parent, SW_{Child} is the weight of the child and α is the weight propagation factor. α is used to maintain the parent-child relationships between the concepts in the user's tree and its value varies between 0 and 1. Al Alshaikh et al. (2017) found that the best value of α is 0.4. Then, all concept weights are divided by the total number of papers that are read by the user in order to normalize the concept weights. The output of this step is a normalized tree of concepts and its corresponding weights for each user.

3.2 Phase 2: Computing the similarity between users and generating CCT

The purpose of this phase is to determine the community of users whose user profiles are similar to the target user. There are three steps in this phase as follows.

3.2.1 Step 1: Find a set of *h* most similar users to a target user

The similarity between a target user and the candidate user is computed using the Tree Edit Distance algorithm (Lakkaraju et al., 2008) to calculate the distance between two DNTC trees (a target user's DNTC and a candidate user's DNTC). This distance is the cost of transforming one tree into another with the minimum number of operations. There are three types of operation: insertion, deletion and substitution. The insertion operation is the cost of inserting a new concept into the tree with a given weight. The deletion operation is the cost of deleting an existing concept with a given weight from the tree. The substitution operation is the cost of changing a concept's weight to another weight. In the 2012 ACM CCS trees we suppose that the concept with zero weight is non-existing node. Hence, the cost of deletion or insertion of a concept is equal to the weight associated with the concept. By contrast, the substitution cost is the difference between weights of an existing concept in both trees. Thus, we calculate the cost of modifying a DNTC tree for a candidate user to match a target user DNTC tree. The two most similar DNTC trees are those which have the lowest total cost of transformations between them. After calculating the total cost between all DNTC trees for candidate users and a target user DNTC tree, the total cost together with its associated id of the user (UserID) are stored as list and these are sorted in increasing order. Hence, the closest candidate user to the target user appears first in the list and the most distant candidate users appear last. Then, the most *h* similar users are selected and stored as set h_i for a target user *i*. *h* is a parameter that will be evaluated in experiments in section 4.2.

3.2.2 Step 2: Generating "Community Centric Tree of Concepts"

The selected *h* similar users are used to generate a Community Centric Tree of Concepts (CCT). The

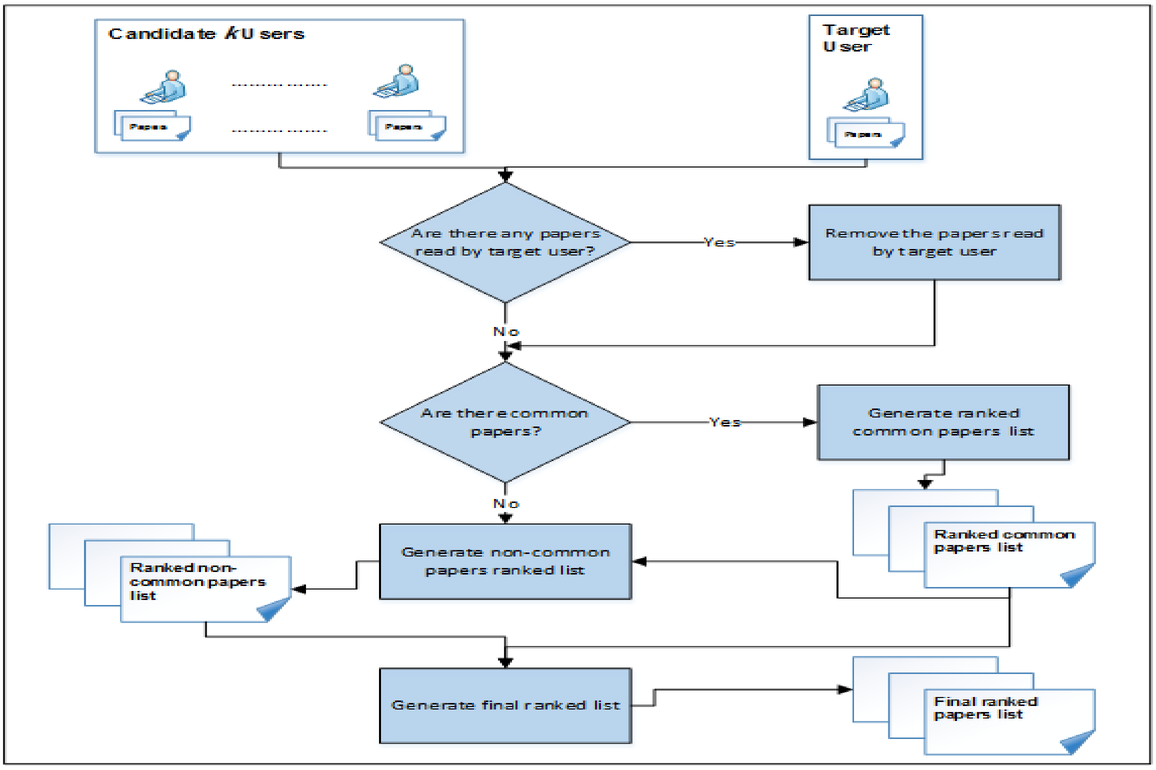

Figure 2: Flowchart for recommendation phase.

CCT is generated by combining the *h* users DNTC profiles as follows. First, *CCTi* for a target user *i* is initialized as tree of 2012 ACM CCS concepts with zero weights for all concepts. Then, the weights for all concepts from all *h* similar users are summing up. Finally, all concept weights are divided by the number of *h* similar users in order to normalize the concept weights. CCT_i represents the centric of the community interests for the target user *i*.

3.2.3 Step 3: Find the *k* most similar users (from the set *h* users)

In this step, we use *CCTi* to find the closest users from the set h_i to the centric of the community interests. The similarity between *CCTi* and the users in the set h_i is computed by using the Tree Edit Distance algorithm. After calculating the total cost between *CCTi* and DNTC trees for the users in the set h_i, the total cost with its associated id of the user (UserID) are stored as a list and sorted in increasing order. Hence, the closest user to *CCTi* appears first and the most distant user appears last. Then, the *k* most similar users are selected and stored as set k_i for a target user *i*. The set k_i is a subset of the set h_i. k_i is a parameter that will be evaluated in experiments in section 4.2. Evaluation results in section 4.2 show that using the set k_i for making recommendations produces better results than using the whole set h_i. This is because the set k_i represents the users that are closer to the CCT_i, which represents the centric of the community interests.

3.3 Phase 3: Recommendation Phase

In this phase, a ranked list of the *top N* research papers is recommended to a target user *i*. First, the papers that are read by the users in the set k_i are retrieved from the database as set Pk_i. If there are any papers already read by a target user *i*, then those papers are removed from the set Pk_i. Then, the set of papers Pk_i is ranked as follows:

a- If some papers appear more than once in the set Pk_i, that means there are common papers between more than one user in the set k_i. The number of appearances of each common paper CP_j in Pk_i is calculated as NCP_j. Then, the papers in Pk_i are ranked according to NCP_j in descending order. Hence, the most common papers have higher ranks. We call this ranked list the common papers list.

b- If there are no common papers (or the common papers are fewer than the number of *top N* recommended papers), then the content-based model is integrated with our collaborative model as follows. We compare the non-common papers profiles with a target user profile. First, a paper profile is represented as tree of concepts as in (Al Alshaikh et al., 2017). Then, the Tree Edit Distance cost is computed between a target user's DNTC tree and the trees of concepts for the non-common papers. We order the papers according to the tree edit distance cost between the paper and the target user's DNTC in increasing order. Hence, the closest papers to a target user appear first and the most distant papers appear last. We call this ranked list the non-common papers list.

The final recommended list that results from the recommendation phase can include both lists: common papers list and non-common papers list. The common papers list appears first before the non-common papers list. Figure 2 shows the flowchart for the recommendation phase.

4 EVALUATION AND RESULTS

In this section, first the evaluation methodology is explained. Then, our model parameters are evaluated to find optimal values. Finally, we compared our proposed model against two baselines.

4.1 Evaluation Methodology

We evaluated the performance of our proposed model using the BibSonomy dataset that contains actual records of users' interests as posts for research papers over approximately a ten-year period. Each post contains: metadata for a research paper, date and time of the post. We consider these posts as users' reading records of research papers. We used users' records for the last two years 2015 and 2016 for users in computing area. This includes 1,642 users and 43,140 research papers. Each paper is classified to the three most closely related concepts from the 2012 ACM CCS ontology. A target user's record is divided into a training set of papers (60%) and testing set of papers (40%). The training set are papers that were read by the user before the testing set. The precision for cut-off results at position N (P_N) is used to evaluate the *top N* recommended papers. The purpose of our paper is to evaluate the future interests/concepts for a target user. Therefore, our precision metric for the future concepts of interest is defined as follows.

Assume a set $FC = \{FC_1, FC_2, \ldots\ldots, FC_m\}$ is a set of future concepts, m is the number of future concepts. A future concept is a concept that does not exist in a target user's training set as shown in Figure 3. The precision for a future concept (FC_i) is defined as follows:

$$P(FC_i)_N = \frac{\text{Number of relevant recommended papers to } FC_i}{N} \qquad (2)$$

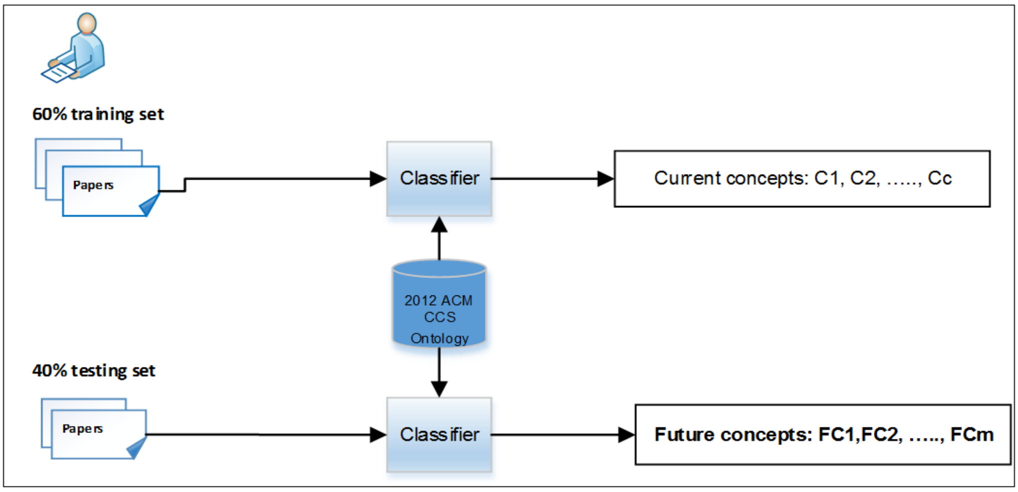

Figure 3: Future concepts.

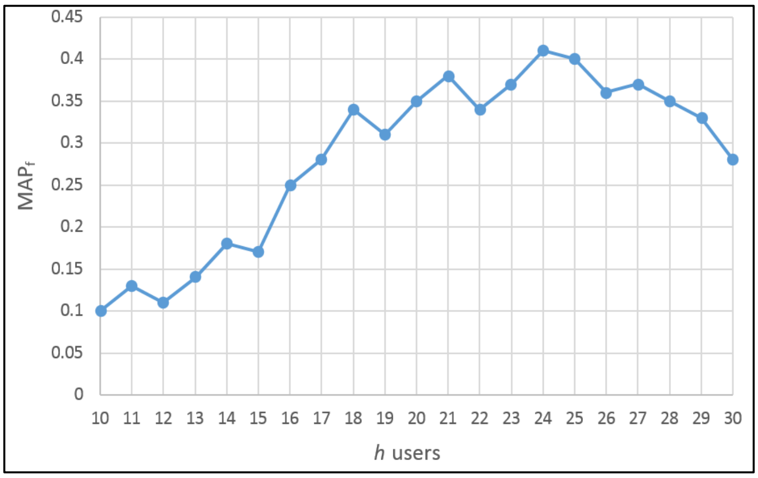

Figure 4: MAP_f results without CCT for different values of *h*.

Then, the average precision (AP_f) for *m* future concepts for a user is calculated as follows:

$$AP_f = \frac{P(FC_1)_N + P(FC_2)_N + \cdots + P(FC_m)_N}{m} \quad (3)$$

The mean average precision for all users is calculated as follows:

$$MAP_f = \frac{\sum_{i=1}^{U} AP_{fi}}{U} \quad (4)$$

where *U* is the total number of users. The top 10 recommended papers are evaluated in our experiments.

Precision is an appropriate type of measurement for systems that only aim at providing highly relevant items to users (Agarwal et al., 2005; Hawalah and Fasli, 2015). Whereas recall and F-measure are not the most appropriate types for these systems for the following reasons. The aim of a research paper recommender system is to present a small amount of relevant information from a massive source of information. Therefore, it is more important to return a small number of recommendations that contains relevant items rather than giving the user a large number of recommendations that may contain more relevant recommendations but also requires the user to select through many irrelevant results. The ratio between the number of relevant results returned and the number of true relevant results is defined as recall. Notice it is possible to have very high recall by making a lot of recommendations. In the research paper domain, a user will be more interested in reading papers that really qualify for his/her interests rather than going through a large list of recommended papers and then selecting those which are of interest. Precision more accurately measures a research paper recommender system ability to reach its aim than recall (Agarwal et al., 2005; Hawalah and Fasli, 2015). Therefore, computing the recall and F-measure usually is not important in a research paper recommender system.

4.2 Evaluating Our Model Parameters

We evaluated our model for two options as follows:

Option1: Without Community-Centric Tree of concepts (Without CCT) (i.e. using the set *h* of users for recommendation phase).

Option 2: With Community-Centric Tree of concepts (With CCT) (i.e. using the set *k* of users for recommendation phase).

First, we have to find the optimal value for *h* in option 1, and optimal values for *h* and *k* in option 2.

Figure 4 shows the MAP_f results of applying our recommender system without CCT. Different values for *h* are tested from 10 to 30 users. It can be clearly seen that the MAP_f results for *h* = 10 are relatively low. This shows that using 10 similar users' papers to be included during recommendation phase is not enough. The MAP_f results increase whenever the *h* value increases until *h*=24. When *h*=24, we have the best result of MAP_f with a score of 0.41. This shows that 24 similar users may hold the most essential concepts that are expected to be related to a target user in future.

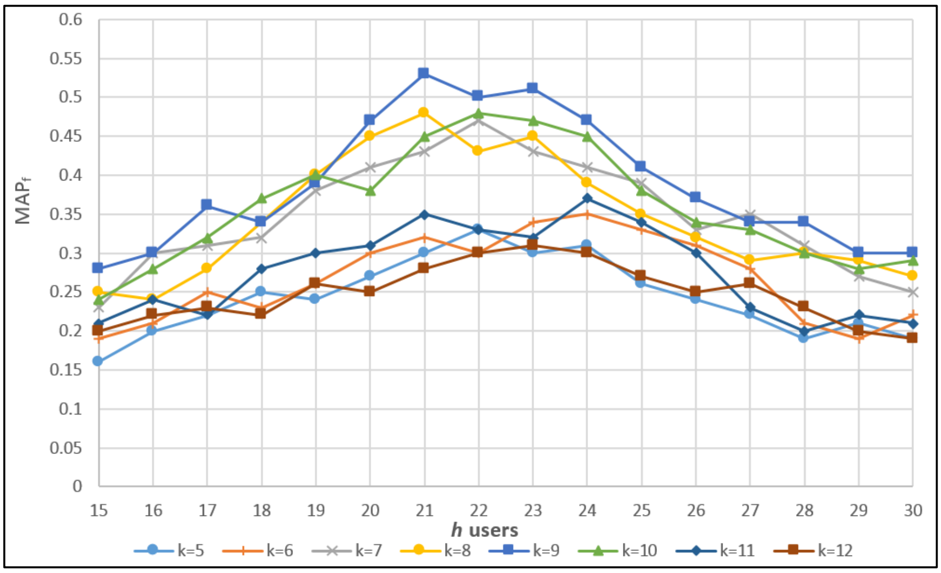

Figure 5: MAP_f results with CCT for different values of *h* and *k*.

Figure 5 shows the MAP_f results of applying our recommender system with CCT using different values for *k* and *h*. We tested our system with different values for *h* from 15 to 30 users. It can be clearly seen that the MAP_f results for *h* = 15 are relatively low. This shows that 15 similar users is a very small number of users to generate CCT using them. The MAP_f results increase whenever the *h* value increases until *h*=21. When *h*=21, we have the best results because 21 similar users may hold the most essential interests to generate CCT. When the *h* value larger than 21, the MAP_f results tend to decrease, this shows that more than 21 similar users is too large number of users to be included when generating the CCT. We tested our system with different values for *k* from 5 to 12 users. The MAP_f results improve when the *h* value comes close to 21 and *k* values increase. The results are very low when *k* = 5, this shows that using only five of the user's papers during recommendation phase is not enough. In general, the best MAP_f results are when *k*=8, *k*=9 and *k*=10. The optimal MAP_f result is 0.53, when *h*=21 and *k*=9.

The results show that the best MAP_f value in option 2 with CCT (MAP_f = 0.53) is greater than the best MAP_f value in option 1 without CCT (MAP_f = 0.41). Therefore, using CCT provides better recommendations in our system.

4.3 Evaluating Our Models against Baselines

We compared our proposed model against two baselines.

Baseline 1: content-based DNTC (Al Alshaikh et al., 2017): a content based recommender system that compares a user's DNTC profile with unread papers' profiles (which are represented as trees of concepts) to recommend the most relevant papers to the target user's interests. The similarity between a target user and a paper is calculated by Tree Edit Distance algorithm.

Baseline 2: User-based Collaborative filtering (UBCF) as in (Nadee et al., 2013): The user-based collaborative filtering model is based on user-item relationships. The similarity between two users is calculated based on the overlap of their paper sets by using the vector cosine similarity algorithm. The *s* most similar users are selected. Then, the missing rating for any paper *i* in target user *a* is predicted by rating the average from the set of *s* users' ratings for paper *i*. The top *N* papers that have the highest average rating from the set *s* similar users are selected to recommend to the target user *a*. To avoid the problem of the lack of user ratings in BibSonomy dataset, we assume that if user *a* did not read paper *i*, then the rating $r_{a,i}$ = 0. If user *a* read paper *i*, then the rating $r_{a,i}$ =1. The BibSonomy system have an

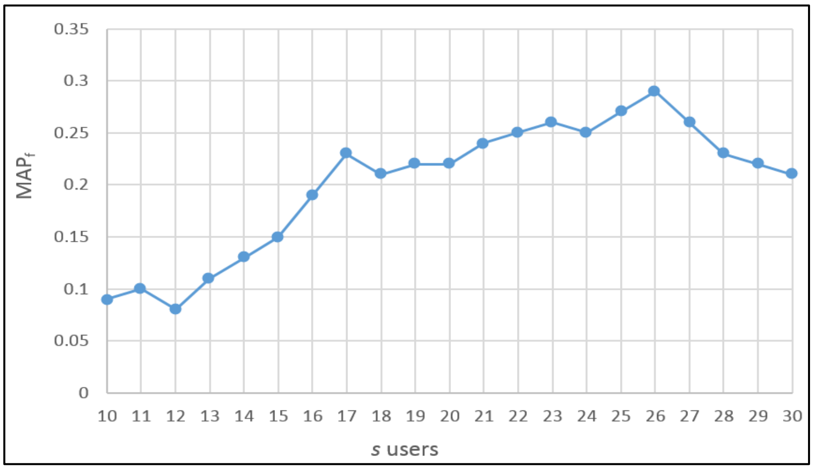

Figure 6: Different values of *s* for UBCF model.

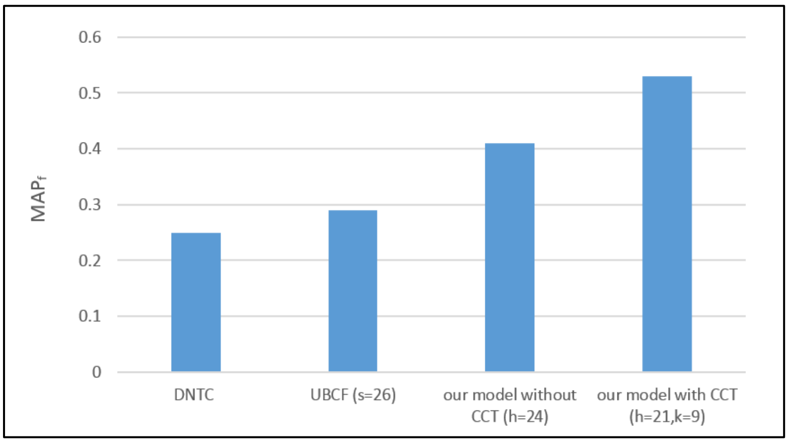

Figure 7: MAP_f results for our model (with and without CCT) against the two baselines.

attribute that indicate if user *a* post paper *i* more than once, hence we assume $r_{a,i} = 2$, if the user post the paper more than once. We tested different values of *s* from 10 to 30 users to find the optimal value of *s*. Figure 6 shows the results for UBCF with different values of *s*. The best MAP_f is 0.29, when $s = 26$.

Figure 7 shows overall comparison results for our system (with and without CCT) against the two baselines. It can be seen that the DNTC model achieves the lowest precision performance with a MAP_f of 0.25. The DNTC model can predict some of user's future concepts because it maintains parent-child relationships between the concepts from the 2012 ACM CCS ontology whilst computing the similarity between a user profile and the new research papers to be recommended. However, DNTC model uses only the current user's interests without considering other potential interests that can be extracted from similar users to the target user.

When it comes to the UBCF model, there is improvement in the performance with MAP_f to 0.29. This model is better than the DNTC model because it considers potential interests that can be concluded from similar users to the target user. However, it has a limitation of sparsity, because UBCF model depends on users rating and the overlap of their paper sets.

Our model (with and without CCT) outperforms the two baselines. This is because it maintains parent-child relationships between the concepts from the 2012 ACM CCS ontology; considers other potential interests that can be extracted from similar users to the target user; and avoids the problem of sparsity. Our model with CCT has better result (i.e. MAP_f =

0.53) than our model without CCT (i.e. $MAP_f = 0.41$). This is because CCT represents the centric of the community interests.

5 CONCLUSIONS

Current content-based recommender systems suffer from overspecialization problem and they may not have the ability to explore potential future interests. Collaborative filtering approaches can solve this problem; however the existing approaches may not be able to locate successful similar users and result in weak recommendations because of the high sparsity problem in the research paper domain. In this paper, we developed a novel collaborative filtering method that does not depend on users' rating. Our novel method computes the similarity between users according to the users' profiles that are represented as Dynamic Normalized Tree of Concepts using 2012 ACM CCS ontology. Then, a Community Centric Tree of concepts (CCT) is generated and used to recommend a set of papers. We performed offline evaluations using the BibSonomy dataset. Different values for the parameters in our model are tested to find the optimal values. Then our model is compared with two baselines: content-based DNTC and User-based Collaborative filtering (UBCF). Our model (with and without CCT) significantly outperforms the two baselines. Our model with CCT has better result than our model without CCT. In future work, we will improve our model to be hybrid approach by including content-based models that are able to detect short-term and long-term user's interests.

REFERENCES

Agarwal, N., Haque, E., Liu, H. and Parsons, L., 2005. Research paper recommender systems: A subspace clustering approach. *In Advances in Web-Age Information Management*, Springer, pp.475-491.

Al Alshaikh, M., Uchyigit G. and Evans, R, 2017. A Research Paper Recommender System Using a Dynamic Normalized Tree of Concepts Model for User Modelling. *In IEEE Eleventh International Conference on Research Challenges in Information Science (IEEE RCIS 2017).* pp.200-210.

ACM Computing Classification System, 2012, URL: https://www.acm.org/about/class/2012.

ACM Computing Classification System, 1998, URL: http://www.acm.org/about/class/1998.

Beel, J., Gipp, B., Langer, S. and Breitinger, C., 2016. Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries*, pp.305-338.

Chandrasekaran, K., Gauch, S., Lakkaraju, P. and Luong, H.P., 2008. Concept-based document recommendations for citeseer authors. *In International Conference on Adaptive Hypermedia and Web-Based Systems*, pp.83-92. Springer Berlin Heidelberg.

Deshpande, M. and Karypis, G., 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, pp.143-177.

Ekstrand, M.D., Riedl, J.T. and Konstan, J.A., 2011. Collaborative filtering recommender systems. *Foundations and Trends in Human–Computer Interaction*, pp.81-173.

Gauch, S., Speretta, M., Chandramouli, A. and Micarelli, A., 2007. User profiles for personalized information access. *The adaptive web*, pp.54-89.

Gordon, G.J. and Singh, A.P., 2008. Relational learning via collective matrix factorization. *In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 650-658.

Hawalah, A. and Fasli, M., 2015. Dynamic user profiles for web personalisation. *Expert Systems with Applications*, 42(5), pp.2547-2569.

Isinkaye, F.O., Folajimi, Y.O. and Ojokoh, B.A., 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, pp.261-273.

Jain, M., 2012. Algorithms for Research Paper Recommendation System. *International Journal of Information Technology*, 5(2), pp.443-445.

Knowledge & Data Engineering Group, University of Kassel: Benchmark Folksonomy Data from BibSonomy, version of January 1st, 2017.

Kotkov, D., Veijalainen, J. and Wang, S., 2016. Challenges of serendipity in recommender systems. *In Proceedings of the 12th International conference on web information systems and technologies.*

Kodakateri Pudhiyaveetil, A., Gauch, S., Luong, H. and Eno, J., 2009. Conceptual recommender system for CiteSeerX. *In Proceedings of the third ACM conference on Recommender systems*, pp. 241-244.

Lakkaraju, P., Gauch, S. and Speretta, M., 2008. Document similarity based on concept tree distance. *In Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, pp. 127-132.

Nadee, W., Li, Y. and Xu, Y., 2013. Acquiring user information needs for recommender systems. *In Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, pp. 5-8.

Sciencepaper Online, 2012, URL: http://www.paper.edu.cn/en.

Shi, Y., Larson, M. and 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.3-48.

Tang, X. and Zeng, Q., 2012. Keyword clustering for user interest profiling refinement within paper recommender systems. *Journal of Systems and Software*, pp.87-101.

Vassiliou, C., Stamoulis, D., Martakos, D. and Athanassopoulos, S., 2006. A recommender system framework combining neural networks & collaborative filtering. *In Proceedings of the 5th WSEAS international conference on Instrumentation, measurement, circuits and systems*. pp. 285-290.
Yang, C., Wei, B., Wu, J., Zhang, Y. and Zhang, L., 2009. CARES: a ranking-oriented CADAL recommender system. *In Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pp. 203-212.
Yang, W., Tang, R. and Lu, L., 2016. A fused method for news recommendation. *In IEEE International Conference in Big Data and Smart Computing (BigComp)*, pp. 341-344.