NEW SIMILARITY MEASURES FOR CAPTURING BROWSING INTERESTS OF USERS INTO WEB USAGE PROFILES

Shaily Kabir

A thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

June 2012

Concordia University

School of Graduate Studies

This is to certify that the thesis prepared

By: Shaily Kabir

Entitled: New Similarity Measures for Capturing Browsing Interests of Users into Web Usage Profiles

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair
Dr. Rajagopalan Jayakumar

_____ Examiner
Dr. Bipin C. Desai

_____ Examiner
Dr. Gregory Butler

_____ Supervisor
Dr. Sudhir P. Mudur

_____ Supervisor
Dr. Nematollaah Shiri

Approved by_____
Chair of Department or Graduate Program Director

_____ 20 _____ _____

Robin A.L. Drew, Ph.D.,ing., Dean

Faculty of Engineering and Computer Science

Abstract

New Similarity Measures for Capturing Browsing Interests of Users into Web Usage Profiles

Shaily Kabir

The essence of web personalization is the adaptability of a website to the needs and interests of individual users. The recognition of user preferences and interests can be based on the knowledge gained from previous interactions of users with the site. Typically, a set of usage profiles is mined from web log data (records of website usage), where each profile models common browsing interests of a group of like-minded users. These profiles are later utilized to provide personalized recommendations. Clearly, the quality of usage profiles is critical to the performance of a personalization system. When using clustering for web mining, successful clustering of users is a major factor in deriving effective usage profiles. Clustering depends on the discriminatory capabilities of the similarity measure used. In this thesis, we first present a new weighted session similarity measure to capture the browsing interests of users into web usage profiles. We base our similarity measure on the reasonable assumption that when users spend longer times on pages or revisit pages in the same session, then very likely, such pages are of greater interest to the user. The proposed similarity measure combines structural similarity with session-wise page significance. The latter, representing the degree of user interest, is computed using page-access frequency and page-access duration. Web usage profiles are generated by applying a fuzzy clustering algorithm using this measure. For evaluating the effectiveness of the proposed measure, we adapt two model-based collaborative filtering algorithms for recommending pages. Experimental results show considerable improvement

in overall performance of recommender systems as compared to other known similarity measures. Lastly, we propose a modification by replacing structural similarity by concept (content) similarity, which we expect would further enhance recommendation system performance.

Acknowledgments

At the commencement, I would like to express my utmost gratefulness to the Almighty Allah for enabling me to complete the dissertation.

I wish to express my deepest gratitude and appreciation to my supervisors Dr. Sudhir P. Mudur and Dr. Nematollaah Shiri for their invaluable guidance and assistance while conducting research and writing my thesis. In particular, they have assisted me to grasp research techniques and to have a solid understanding of how to draw research questions and dig into the research area. Further, my sincere thankfulness goes to them for their insightful directions, necessary revisions, and constructive suggestions on the preparation of a paper, which we co-authored, for the 10^{th} Workshop on Intelligent Techniques for Web Personalization and Recommendation, 2012 (ITWP'12).

I would like to thank my student colleagues and friends for lending their hands and sparing time in needs. Surely, I would remember the time spent together and would recall the happiness, sorrows, and mutual cooperation during my journey in Concordia.

I owe to my sister Upama Kabir, my brother-in-law Mosaddek Hossain Kamal, my niece Tulona, and my friends Mosarrat Jahan and Rajneesh Singla for their continuous support, care, and encouragement during last two years.

Last but not the least, it would have been difficult on my part to accomplish this work without continuous support and sacrifice from my beloved parents and also from my husband, Chowdhury Ziauddin Hayat. They have missed my company and affection due to my studies in Canada. Their unending love and uncompromising faith always inspired me and bolstered my confidence throughout my journey abroad.

Contents

List of Figures **ix**

List of Tables **xi**

1 Introduction **1**

1.1 Research Motivation 4

1.2 Research Problem 6

1.3 Methodology 6

1.4 Contributions 7

1.5 Outline of the Thesis 8

2 Background and Related Works **10**

2.1 Web Usage Mining (WUM) 10

2.1.1 Data Preprocessing 11

2.1.2 Pattern Discovery 14

2.1.3 Pattern Analysis 17

2.1.4 Applications of Web Usage Mining 17

2.2 Relational Fuzzy Subtractive Clustering (RFSC) Algorithm 18

2.3 Web Recommender Systems and Web Usage Mining 20

2.3.1 Content-Based Filtering 21

2.3.2 Manual Rule-Based Filtering 22

2.3.3 Collaborative Filtering 22

2.4 Similarity Measures 26

3 Preprocessing - Creation of Weighted Sessions 31

3.1 Recommender System Architecture 31

3.2 Usage Session Extraction 33

3.3 Degree of User Interests 34

3.4 Weighted Usage Session Conversion 35

4 Weighted Session Similarity Measure 37

4.1 Web Directory Structure 37

4.2 URL Similarity Measure 38

4.3 Weighted Session Similarity Measure 44

4.4 Usage Profile Generation 47

5 Recommender Algorithms 48

5.1 A Brief Review of Collaborative Filtering(CF) Algorithms 48

5.1.1 Memory-Based Collaborative Filtering(CF) Approach 49

5.1.2 Model-Based Collaborative Filtering(CF) Approach 50

5.1.3 Hybrid Collaborative Filtering(CF) Approach 51

5.2 Adaptation of Recommender Algorithms 52

5.2.1 Model-Based CF Algorithm with Similarity and Overlapping Ratio . 54

5.2.2 Fuzzy Hybrid CF Algorithm with Similarity and Overlapping Ratio 54

6 Experimental Results and Performance Evaluation 56

6.1 Dataset and Experimental Setup 56

6.2 Performance Evaluation Metrics 58

6.3 Performance Analysis 59

6.3.1 Performance Analysis for the Most Significant Hidden_set 61

6.3.2 Performance Analysis for the Randomly Selected Hidden_set 68

6.3.2.1 Performance Analysis of Recommending the *High-Significant* Pages 69

6.3.2.2 Performance Analysis of Recommending the *High-Ranked* Pages . . . 75
6.3.3 Impact of Number of Nearest Clusters . . . 78
6.3.4 Impact of DSR in the Modified Fuzzy Hybrid Collaborative Filtering Algorithm . . . 80
6.3.5 Impact of Overlapping Ratio in Modified Model-based and Fuzzy Hybrid Collaborative Filtering Algorithms . . . 81

7 Semantic Inclusion to Weighted Session Similarity Measure 83
7.1 Concept Hierarchy . . . 83
7.2 Proposed Semantic Similarity Measure (SESM) Between Web Pages . . . 85
7.3 Incorporation of SESM into Weighted Session Similarity Measure . . . 89

8 Conclusions and Future Work 91

Bibliography 93

List of Figures

1 Web Usage Mining Process. 11

2 A General Framework of WUM-based Personalization System [36]. 25

3 The Dataflow Diagram of Our Personalization System. 32

4 A Part of Page Hierarchy of a "University CS Department" Website. 38

5 A Part of Page Hierarchy of "RBC Royal Bank" Website. 42

6 Comparison of *Recall*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 63

7 Comparison of *Precision*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 63

8 Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 66

9 Comparison of *Precision*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 67

10 Comparison of *Recall*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 71

11 Comparison of *Precision*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 72

12 Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 73

13 Comparison of *Precision*(%) when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 74

14 A Part of Concept Hierarchy of “CS Department” Website. 85
15 A Part of Concept Hierarchy of “CS Department” Website with Access-count. 87
16 A Part of Concept Hierarchy of “CS Department” Website with Probability. 88
17 A Part of Concept Hierarchy of “CS Department” Website with Weight. . . 88

List of Tables

1 Example of Entries in a Typical Web log in Common Log File format. . . . 12

2 The Common Log File format. 13

3 URL similarity matrix for Fig. 4 using equation 9 [40]. 40

4 URL similarity matrix for Fig. 4 using our proposed similarity measure. . . 41

5 URL similarity matrix for Fig. 5 using equation 9 [40]. 42

6 URL similarity matrix for Fig. 5 using our proposed similarity measure. . . 43

7 Total number of clusters for 10,864 training weighted sessions (Case-1). . . 59

8 Total number of clusters for 9506 training weighted sessions (Case-2). . . . 59

9 Total number of clusters for 12,222 training weighted sessions (Case-3). . . 59

10 Required time (in hours) for the usage profile generation from 10,864 weighted sessions (Case-1). 60

11 Two most prominent usage profiles of UP_1, UP_2, and UP_3 for 10,864 weighted sessions (Case-1). 60

12 Two most prominent usage profiles of UP_4 and UP_5 for 10,864 weighted sessions (Case-1). 60

13 Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 61

14 Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2). 61

15 Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3). 61

16 Comparison of *MRHR*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 62
17 Recommendaiton time(in seconds) per user for modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with *top_N*=20. 62
18 Comparison of results when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2). 64
19 Comparison of results when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3). 64
20 Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 64
21 Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2). 65
22 Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3). 65
23 Comparison of *MRHR*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 66
24 Recommendaiton time(in seconds) per user for modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with top_N=20. 66
25 Comparison of results when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2). 67
26 Comparison of results when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3). 67
27 Comparison of overall *Hits* when using modified model-based and fuzzy CF algorithms for randomly selected *Hidden_set* from 2716 test sessions (Case-1). 69
28 Comparison of results when using modified model-based CF algorithm for randomly selected *Hidden_set* from 2716 test sessions (Case-1). 69

29 Comparison of results when using modified fuzzy hybrid CF algorithm for randomly selected *Hidden_set* from 2716 test sessions (Case-1). 69
30 Comparison of *Hits* for 1122 randomly selected *high-significant* hidden pages (Case-1). 70
31 Comparison of *Hits* for 1638 randomly selected *high-significant* hidden pages (Case-2). 70
32 Comparison of *Hits* for 554 randomly selected *high-significant* hidden pages (Case-3). 70
33 Comparison of *MRHR*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 71
34 Comparison of results when using modified model-based CF algorithm for 1638 randomly selected *high-significant* hidden pages (Case-2). 72
35 Comparison of results when using modified model-based CF algorithm for 554 randomly selected *high-significant* hidden pages (Case-3). 73
36 Comparison of *MRHR*(%) when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1). 74
37 Comparison of results when using modified fuzzy hybrid CF algorithm for 1638 randomly selected *high-significant* hidden pages (Case-2). 75
38 Comparison of results when using modified fuzzy hybrid CF algorithm for 554 randomly selected *high-significant* hidden pages (Case-3). 75
39 Comparison of *Recall*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1). 75
40 Comparison of *Precision*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1). 76
41 Comparison of *MRHR*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1). 76
42 Comparison of results when using modified model-based CF algorithm for 2476 randomly selected *high-rank* hidden pages (Case-2). 77

43 Comparison of results when using modified model-based CF algorithm for 811 randomly selected *high-rank* hidden pages (Case-3). 77
44 Comparison of results when using modified fuzzy hybrid CF algorithm for 2476 randomly selected *high-rank* hidden pages (Case-2). 77
45 Comparison of results when using modified fuzzy hybrid CF algorithm for 811 randomly selected *high-rank* hidden pages (Case-3). 77
46 Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with NP=1 and 2. 78
47 Comparison of *Hits* when using modified model-based CF algorithm for the randomly selected *Hidden_set* from 4076 test sessions (Case-2) with NP=1 and 2. 78
48 Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with *Nearest_K*=100 and 200. 79
49 Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the randomly selected *Hidden_set* from 1358 test sessions (Case-3) with Nearest_K=100 and 200. 79
50 Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for *high-significant* pages from 2716 test sessions (Case-1) with DSR=0.10 and 0.05. 80
51 Comparison of recommendation time(in seconds) per user when using modified fuzzy hybrid CF algorithm for *high-significant* pages from 2716 test sessions (Case-1) with DSR=0.10 and 0.05. 80
52 Impact of similarity measure and overlapping ratio in modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1). 81

53 Impact of similarity measure and overlapping ratio in modified fuzzy hybrid CF algorithm for randomly selected *Hidden_set* from 4074 test sessions (Case-2). 82

Chapter 1

Introduction

The World Wide Web has been considered as the most dominant source of information, with which most people interact these days. According to the survey conducted by the site, Pingdom[1] by December 2001, there were approximately 555 million websites and 2.27 billion web users (almost doubled in just five years). This exponential rise in web usage further accelerates the extent of information stored in the web. This explosive growth of web information however lacks an integrated structure or schema and hence poses difficulties for the users in discovering relevant information quickly.

An increasing demand for web-based services such as e-commerce, e-banking has changed the way the web is now being used. Business entities use the web for their business purposes, and provide lot of information and advertisements to draw attention of users, which make the whole web environment highly competitive. From a business point of view, web-service providers would want to retain their previous customers (i.e., users), attract potential new ones, and in general convert them into loyal customers, rather than just casual visitors to their website. However, users are often overwhelmed with the vast amount of information they have to wade through, which leads to frustration, a phenomenon known as "information overloading". These two interrelated issues can be dealt with by providing personalized services to the users by realizing their needs and interests, and directing them to relevant information. In other words, adapting a website through personalization can improve user

[1] http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/

interactions with the website, which in turn will enhance user retention and loyalty.

A personalized website typically recognizes user preferences and needs, and adapts it's services to assist users find relevant information quickly. Thereby, it offers a friendly web environment to individual users. Further, it develops a trustworthy relationship between the website and the users. Generally, search engines help users to navigate and find relevant information pages. However, in a large website with thousands of dynamic web pages, the users may need some guidance for navigating further within the website, and to efficiently find the information which they are seeking for. Although in many large websites there are searching options via keywords, personalization automatically provides suggestions to the users by recognizing their browsing interests, without explicitly asking them. In summary, personalization makes user interactions with the website easier, saves time, and helps create loyal users-one of the main goals of any website. In other words, to a great extent, it alleviates the information overload problem of the users.

People often muddle up two terms, customization and recommendation in the context of web personalization. Intuitively, customization is user-driven, whereas recommendation is system-driven. In customization, users have to first configure the structure and layout of the website. In subsequent visits, the personalization system then presents them with web pages according to their preferences. An example of such a personalization system exploiting customization is Yahoo!, which offers customized features to the users in order to create personalized "MyPortal" site. In recommendation, the personalization system dynamically monitors and analyzes user behavior while browsing through the website, and eventually generates recommendations of interest to the user. An example of such a personalization system is the WebPersonalizer [34].

Web recommender systems, tools for web personalization, recommend a list of items that are potentially preferred by the users. The recommendation problem can be defined as an estimation of preferences for the items that have not yet been seen by the users so as to able to recommend them. Items can be any type of online information resources such as web pages, videos, musics, and books. They are beneficial for e-commerce in increasing

potential customers and sales by providing users with more interesting options to browse and more items to assess. Suggestions for relevant books on Amazon[2], or movies on Netflix[3] are real-world examples of commercial recommender systems. Typically, these systems are implemented on the web server, and their designs particularly depend on the application domains and the characteristics of the available data.

Input to recommender systems can be classified into three categories: user ratings, demographic data, and content data. The implicit and explicit users' ratings on pages/items represent their interests or opinions on the items. Explicit ratings are normally provided using a specific numerical scale. For example, the users of Netflix give their ratings on movies using a scale between 1 ("dislike") and 5 ("like"). In some cases, a binary rating scheme is used, where the rating is either 0 or 1. User's ratings can also be gathered implicitly from the web log data, i.e., user purchase history or types of information access patterns. Besides, demographic data, such as age, gender, and education of the users, can also serve as a source of input to the recommender systems. But without explicit user input, these data are usually difficult to assemble. Content data can be used as an alternative input source. Content data relies on a textual analysis of documents rated by the users. Generally, features extracted by this analysis are used in the recommender systems for deriving user profile. Further, output of the recommender systems can be classified into two categories: prediction and recommendation. In prediction, the recommender systems predict a particular item that is likely to be requested subsequently by the user. In recommendation, the recommender systems suggest a list of *top-N* items that is of potential interest to the user.

There are three basic approaches, termed as filtering, used in recommender systems for automatically providing recommendations to users [15, 39]. These are content-based filtering, manual rule-based filtering, and collaborative filtering. However, collaborative filtering, the approach pursued in our research as well, is the most successful and widely used approach in commercial recommender systems. It provides recommendations to a target user (i.e., an active user) based on known/derived preferences of other users. The

[2]www.amazon.com
[3]www.netflix.com

key assumption in collaborative filtering is that an active user will prefer those items that are chosen by like-minded users, or even not preferred by dissimilar users [66]. In general, this approach stores the preferences and the opinions of all previous users, analyzes historic data, and generates recommendations to the active user by finding a group of users having similar interests and opinions. A more detailed review of these different approaches is presented in the next chapter.

1.1 Research Motivation

The traditional collaborative filtering (CF) technique, also known as memory-based CF technique [7, 19, 66], utilizes the entire database of ratings on items by all past users to make recommendations. The ratings on items can be achieved explicitly from the users, or implicitly by observing users browsing activities, their purchases etc. The current ratings of an active user is then matched in real time against the rating-database in order to discover a set of neighbor users, who have historically similar preferences to the active user. It is likely that the active user will have preferences for the items that are similar to those of his neighbors. Therefore, by combining the ratings of the selected neighbors, it is possible to predict items for the active user. The memory-based CF approach is used in many recommender systems such as GroupLens [45], Ringo [53], and others, due to its simplicity and ease of implementation. However, the memory-based CF systems have two fundamental limitations - difficulty in handling data sparsity and scalability.

In a large website, users rate only a small portion of items, instead of the whole set of items. This makes the entire user-item rating matrix highly sparse in nature. Generally, similarity among user ratings in memory-based CF techniques is based only on common items, but the common items are relatively very few in numbers. This often makes similarity result unreliable. Consequently, a very poor recommendation accuracy is achieved. In addition, the computational complexity (i.e., space and time complexity) of the memory-based CF approach is linearly amplified with increasing number of users and items, making the system less scalable as the websites become larger. For reducing these limitations and

for achieving better recommendation performance, model-based CF techniques [7,35,48,66] have been proposed. The model-based CF approach utilizes the user-rating database in order to learn a model offline and then makes use of this model to provide recommendations to active users in real time. In this context, various web usage mining techniques such as clustering, association rule generation and sequential pattern discovery have been applied. Web usage mining is defined as an application of data mining techniques to the web data for discovering interesting information about the user navigation behaviors [56]. The mining techniques are applied on historic-usage-data in order to build up an access behavior model, representing the variety of user access patterns in the website [35, 39]. The advantages of the model-based CF approach is that it reduces online processing time as matching of the active user is now done with respect to the access behavior model instead of the whole database, thereby mitigating the scalability problem as well. However, it should be noted that this increasing scalability of the model-based CF system may often result in reduced recommendation accuracy.

The performance of a model-based CF system largely depends on the quality of the underlying model, used to provide recommendations. Typically, the access behavior model can be generated in the form of a set of usage profiles by clustering users based on their similar interests [35, 36, 40, 48, 59, 66]. Each usage profile captures the common interests of a group of users accessing the website. The usage profiles can successfully be utilized in making recommendations, predicting browsing path, pre-fetching pages, better restructuring the website, i.e., improving user experience while browsing, etc. It is important to mention that the quality of usage profiles directly influences the recommendation performance. Better quality of usage profiles assists in getting higher recommendation quality. However, the quality of profiles relies on how effectively the web users are grouped together based on their interests.

1.2 Research Problem

We can now state our research problem as follows:
Given historic usage data of a website in the form of web logs, develop methods which enable recommender systems to take into account the browsing interests of users by using information which is implicitly present in the web log data. The methods developed should be efficient and scalable.

1.3 Methodology

As already mentioned earlier, clustering is one of the most popular mining techniques used in developing usage profiles which form the principal component of recommender systems following the model-based CF approach. Successful clustering of users is the key to generate effective usage profiles. On the other hand, clustering depends very much on the similarity measure defined among the users/items to be clustered. In the absence of explicit user ratings, the accuracy of similarity computation largely depends on how closely the implicit interests of users can be measured from the user browsing data. Our methodology is based on the following. A user's implicit interest in a page can be inferred to a reasonable extent from the time duration spent on the page and/or the frequency with which a page has been accessed. Usually, longer time spent or more frequent access of a page indicates higher user interest in that page. A page which is more interesting to a user indicates more significance to the user. In general, two users are said to have similar taste when they are interested on the same item(s). In contrast, two users' interest on the same item may differ due to varying page-access frequency and/or different page-viewing time. In addition, two users may be interested in pages, which are not identical but may be conveying similar information (same topic or subject). Therefore, we believe that it is better to consider all these aspects when defining a similarity measure among users. Notably, the similarity measure also plays a critical role in recommendation performance as it also helps select the nearest usage profile(s) for the active user in real time.

Previous work on usage session similarity mostly takes into account the page structure similarity, i.e. locations of pages in the website with respect to one another. Our focus is on defining a similarity measure among users which can help capture user browsing interests as well. In defining a new similarity measure, we will use both browsing interest, denoted henceforth as page significance and the structural similarity among pages. Page significance will be derived from web log data. The similarity measure will then be used in a fuzzy clustering technique [59] to generate a set of usage profiles, representing user access patterns on that website. The similarity measure and the usage profiles generated will be used to adapt recommender systems using the model-based CF approach. For evaluating the effectiveness of our similarity measure, we will compare our results against those obtained using other popular similarity measures, namely, Pearson correlation-coefficient, Cosine similarity measure, Jaccard coefficient and the similarity measure proposed by Nasraoui et al. [40].

1.4 Contributions

Our major contribution is a new weighted similarity measure for effectively capturing browsing interests of users from web log data, and its application in clustering, usage profile generation and recommendation systems. We convincingly demonstrate this via extensive experiments and comparison with other popular similarity measures used in web mining.

The details of our contribution are as follows:

1. We introduce a new page-significance measure for estimating user interest in a page, taking into account the time spent on the page and multiple page visits.

2. Next, we present a modification to URL-based similarity measure, based on the locations of pages in a website page hierarchy, satisfying two important aspects of the page hierarchy: more specialized information is normally present in lower-level nodes, and content in the urls is conceptually more related when going deeper into the page hierarchy.

3. Our most important contribution is the definition of a new weighted similarity measure which computes similarity among usage sessions by utilizing both session-wise page significance and their structural similarity. The page significance captures user interests, and structural similarity incorporates similarity of the topics in the pages.

4. For improving the performance of the model-based CF method for recommendation, we introduce a new parameter named "overlapping ratio". This is the ratio of common items between the active user and the cluster prototypes. For selecting the nearest cluster for the active user, we jointly apply this parameter and the similarity value between the active user and the prototypes, which results in improved recommendation hits in all the cases we have experimented with.

5. We adapt two model-based CF methods in order to evaluate the effectiveness of the proposed similarity measure and compare it with other popular measures. First is the model-based CF with similarity and overlapping Ratio, and the second is the fuzzy hybrid CF with similarity and overlapping ratio.

6. Lastly, we propose an important modification to incorporate semantics into our weighted similarity measure by employing conceptual relationship among pages.

1.5 Outline of the Thesis

The rest of this thesis is organized as follows.

In Chapter 2, we present an overview of web usage mining (WUM), an overview of the selected fuzzy clustering approach, the incorporation of the WUM in web personalization, and a comprehensive review of related work on similarity measures and their utilization in creating web usage profiles through clustering.

In Chapter 3, we present our proposed WUM-based personalization system architecture, and web data preprocessing including the new page-significance measure and conversion of usage session data into weighted sessions.

In Chapter 4, we give a definition of the new weighted session similarity measure together

including the modified URL-based page similarity measure and usage profile generation through fuzzy clustering.

In Chapter 5, we introduce a new parameter "overlapping ratio" and discuss its use in our adaptation of two model-based CF algorithms.

In Chapter 6, we describe the various experiments conducted, analyze the results and demonstrate the effectiveness of our weighted similarity measure with the help of various performance evaluation metrics.

In Chapter 7, we present the modification to incorporate conceptual (semantic) similarity among pages into our weighted session similarity measure. Finally, concluding remarks and possible directions for future research are given in Chapter 8.

Chapter 2

Background and Related Works

In this chapter, we study web usage mining (WUM) process along with its integration in web personalization systems. Firstly, we introduce the WUM process and it's applications. Next, we discuss the Relational Fuzzy Subtractive Clustering (RFSC) algorithm, used in our work for grouping sessions. Following this, we describe web personalization systems based on the WUM. Lastly, we review related work on usage session similarity measures used in clustering and usage profile generation, later utilized by a personalization system to make recommendations.

2.1 Web Usage Mining (WUM)

Web usage mining (WUM) is the application of data mining techniques to web data with the goal of discovering interesting information about user navigation behaviors [56]. Typically, what is discovered are usage patterns represented as collections of pages, that are frequently accessed by groups of users with common needs and interests. These patterns are particularly useful for many web-based applications including web personalization, system performance and web traffic analysis, website modifications, e-commerce, and etc. Fig. 1 presents the overall web usage mining process. Generally, the whole process is partitioned into three phases.

1. Data collection and preprocessing,

2. Pattern discovery, and

3. Pattern analysis

We discuss these phases in following subsections.

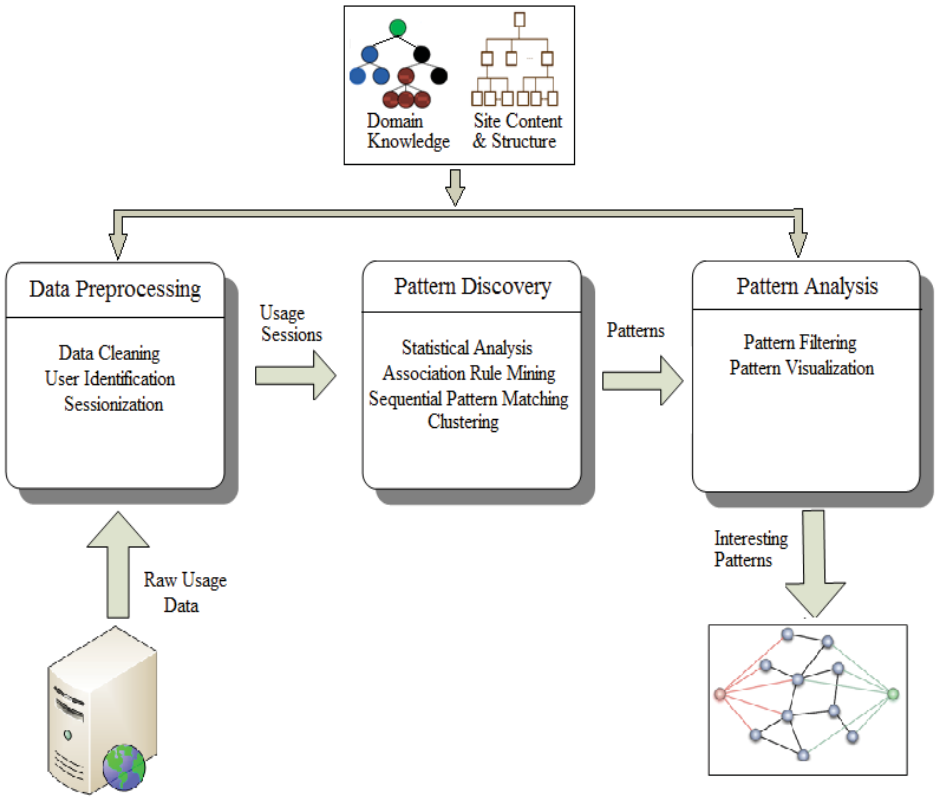


Figure 1: Web Usage Mining Process.

2.1.1 Data Preprocessing

A web server log is the primary source of usage data for web usage mining. Generally, the log files store all browsing activities of web users. However, reliability issues arise with the web log due to various levels of caching present in the web environment. An important point to note is that cached pages are not stored in the log. Hence, better quality and reliable usage information can be obtained by combining the usage information from the web log with navigation information from other sources such as client-side, proxy server, and etc.

At the client side, the user browsing behaviors can be captured in detail by using a remote agent (such as Javascript or Java applet) or by modifying the client browser. However, client-side data collection is limited due to privacy issue, and without user cooperation it is difficult. Information available in proxy-caching can also be utilized as a source, but this information usually characterizes a group of anonymous users sharing the same proxy server [56]. Other than these sources, information retrieved from the content and structure of the website, domain knowledge, and etc. also play essential roles in data preprocessing and pattern discovery phases.

Typically, the web log records the history of all page requests made by the users. There are two standard formats for keeping log records: Common Log File (CLF) format and Extended Log File (ELF) format. Table. 1 represents a fragment of a typical web log file in Common Log File format, and Table. 2 describes the essential fields in this format. Note that, the web log with ELF format maintains all fields as in CLF format with two additional fields, user-agent field (i.e., the browser type used to access) and referrer field (i.e., the previous URL that referred the user to the resources requested).

Information available in the web log are too raw for the pattern discovery phase. They need further processing so that after the data preprocessing, they are presented in a way, suitable for use in the mining process [13]. The preprocessing mainly includes data cleaning, user identification, and session reconstruction. In data cleaning, irrelevant log entries such as the entries related to image files (i.e., .gif, .jpeg, and etc.), requests from web robots (also known as spiders or crawlers), and the entries with unsuccessful HTTP status code are removed. The next step is to identify individual user activities, and to group

Table 1: Example of Entries in a Typical Web log in Common Log File format.

202.161.108.167	-	-	[01/Feb/2003:00:00:03 +1100]	"GET Image1.gif HTTP/1.1"	200	14102
213.183.13.65	-	-	[01/Feb/2003:00:00:16 +1100]	"GET A.html HTT P/1.1"	200	244
66.249.65.107	-	-	[01/Feb/2003:00:04:10 +1100]	"GET B.html HTTP/1.1"	200	11179
172.21.13.45	-	-	[01/Feb/2003:00:04:12 +1100]	"GET C.html HTTP/1.1"	200	3401

Table 2: The Common Log File format.

Fields	Description
remotehost	Remote host-name (or IP number).
logname	The identifier used to identify the client making the HTTP request. If no value is present, a "-" is substituted.
authuser	The user-name (or user ID) used by the client for authentication. If no value is present, a "-" is substituted.
date	The date, time, and time-zone when the server finished processing the request.
request	The request line came from the client specifying the method, requested resource, and the protocol of the request.
status	The status code indicating the success or failure of the HTTP request, where 2xx is a successful response, 3xx a redirection, 4xx a client error, and 5xx a server error.
bytes	The content-length of the document transferred, not including the HTTP header.

all browsing activities related to individual users, thus generating an activity-log for each user anonymously. In the absence of user authentication mechanism, the user IP address is used to group the activities. However, IP address is not always reliable due to the proxy server. Therefore, grouping individual user activities requires integrating with other available information (i.e., browser or referrer information) with user IP address [23]. Once the individual user activities are grouped, the next task is to reconstruct a set of sessions for each individual user. Basically, the session reconstruction is a partitioning of the usage activity-log in a meaningful way. Each session is defined as a list of user browsing activities, while visiting the site (i.e., the moment he/she enters the website until the time he/she leaves it). This reconstruction process is called as sessionization. Typically, sessionization is essential for analyzing usage behavior in the web. A number of sessionization heuristics have been evolved, which can be classified into two main categories: navigation-oriented heuristic and time-oriented heuristic [4]. Navigation-oriented heuristics exploit behavioral

habits associated with web navigation, for example, a request for a page that is unreachable through the pages visited so far is likely to have been initiated by another user. The time-oriented heuristic can be two types: the session-duration heuristic and page-stay time heuristic. These two time-oriented heuristics can be used individually or jointly to segment the user activity log into sessions. Generally, the session-duration varies from 25.5 minutes [9] to 24 hours [67], while 30 minutes is the mostly used default timeout for session duration [4, 5, 28, 54, 70]. Page-stay time varies with respect to the page content and the nature of the applications. In general, a 10-minute cutoff has been adopted as the page-stay time in literature [4, 5, 28, 54, 70]. Besides, the navigation-oriented heuristic depends on availability of the referrer information, and can be used individually or combined with the time-oriented heuristic for sessionization.

At the end of sessionization, a set of M pages, U=$\{\text{url}_1,\text{url}_2,...,\text{url}_M\}$ and a set of N sessions US=$\{\text{us}_1,\text{us}_2,...,\text{us}_N\}$ are received from the web log, where each usage session us_i is a subset of U. These serve as input to the pattern discovery phase. Each session can be represented as a set or a sequence of accessed pages. However, this largely depends on the web usage mining application. In some cases such as user navigation-path analysis, it is required for a session to be in the form of an ordered sequence of pages. However, many applications such as market basket analysis and usage profile analysis could use a session as a set of accessed pages, without access ordering. In our work, we consider each usage session as a set of accessed pages.

2.1.2 Pattern Discovery

The pattern discovery phase applies different methods and algorithms from a variety of fields such as statistics, data mining, machine learning, and pattern recognition on the pre-processed usage data for discovering meaningful and interesting usage patterns of the web users [56]. We discuss next these techniques in the context of web mining.

- Statistical Analysis

  In statistical analysis, information about the user behavior can be extracted by applying different statistical techniques such as mean, median, frequency on different data items such as accessed pages, access rate, visiting time, and length of a navigational path. The extracted knowledge is particularly helpful in potential improvement of system performance, facilitating the website modification, and etc.

- Association Rule Generation

  In association rule generation, groups of pages, commonly accessed together in most of the sessions, are discovered. Pages in the discovered groups may not be associated to each other via hyperlinks. A well-known example of such mining technique is the Apriori algorithm [3]. While employing the Apriori algorithm in business and marketing applications, it may discover, for example, correlation among the users interested in the information of electronic products to those users interested in sporting equipments information. Besides this, the association rule can be used in restructuring the site or in pre-fetching pages from a remote site.

- Sequential Patterns Analysis

  In sequential patterns analysis, groups of pages are discovered that are accessed sequentially in a time-ordered set of sessions. The discovered patterns are useful in placing advertisements to certain user groups by predicting future visit patterns.

- Clustering

  Clustering is a partitioning of the web pages/users into a number of groups based on their similarity/dissimilarity. Each group is called as a cluster. In each cluster, objects are similar to each other, and at the same time dissimilar to the objects in other clusters. Therefore, the goal of clustering is to maximize the intra-group similarity and to minimize the inter-group similarity. If we consider a dataset US=$\{us_1,us_2,...,us_N\}$ of N of usage sessions, then clustering divides US into Q groups $\{C_1,C_2,....C_Q\}$ with the following constraints:

a. $C_Z \neq \emptyset$ for Z=1,2,.... ,Q;

b. $\bigcup_{Z=1}^{Q} C_Z = US$;

A wide variety of clustering algorithms have been proposed in the literature including k-means, leader, hierarchical, etc.

The result of clustering is presented using a membership matrix MV[Q×N], where each entry MV[C_T,F] shows a membership value of us_F in cluster C_T. For a crisp clustering (also known as hard clustering), each usage session is a member of only one cluster with a membership value of 1; for other clusters, it's membership value is 0. However, in fuzzy clustering (also known as soft clustering) each usage session is a member of every clusters with varying degree of membership value, within the range of 0 and 1.

Typically, the goal of clustering in web usage mining is to develop a usage model as a set of usage profiles, used in many Web-based applications. These profiles are generated from the patterns discovered by the clustering process. There are two objects used for clustering in web mining: sessions (or users) or web pages. Each of these is useful in different applications, and in particular, both clustering results can be used for Web personalization. In usage session clustering, users with similar browsing preferences are grouped, which is useful for e-commerce applications, and for providing personalized information to the users. On the other hand, in page clustering, pages having related contents are grouped together, which is helpful for search engines and web assistance providers. Another factor is whether one should use crisp or fuzzy clustering. According to many web usage mining practices, it is better to use fuzzy clustering approach so as to be able to deal with the fuzziness and uncertainty in web usage data [40]. In our work, we use the RFSC algorithm [59], a fuzzy clustering method, to group the sessions for generating a set of usage profiles, where users with similar access preferences belong to the same profile. The RFSC algorithm is discussed in greater detail in section 2.2.

2.1.3 Pattern Analysis

Once access patterns have been discovered, they are evaluated and usually presented in a form that is understandable to humans, or input to visualization techniques using appropriate tools and techniques. Examples of such tools include the WebViz system for visualizing the path traversal patterns. Others have proposed pattern analysis tools using OLAP techniques for simplifying the analysis of usage statistics from web server logs. Generally, pattern analysis methodology depends on the application for which the web usage mining has been carried out. For web personalization, the extracted patterns are usually incorporated into a personalization system.

2.1.4 Applications of Web Usage Mining

The basic goal of web usage mining (WUM) is to discover interesting usage patterns of the website content by the users. The discovered information is exploited later by a number of web-based applications. The major applications of web usage mining are briefly discussed below:

- Web Personalization

  Web usage mining can facilitate a system to provide personalized web usage experience. To do personalization, web recommender systems are most commonly used. A recommender system attempts to predict user preferences by matching their access behavior with the usage patterns discovered by the WUM, and to ease their navigation experiences by suggesting a list of pages, which are likely to be preferred by them. For instance, WebWatcher, SiteHelper, the clustering work by Mobasher et. al. [36], and etc., are used for providing website personalization based on usage information [56]. Section 2.4 discusses in detail another application of web usage mining, personalizing the site.

- Pre-fetching and Caching

  Web usage mining can be used to develop proper pre-fetching and caching strategies by understanding the nature of web traffic. These strategies effectively reduce web server

response time. In turn, the performance of web server and web-based applications is improved, enhancing user satisfaction of the site.

- Website Modification

  The usage patterns obtained from web usage mining can provide basic guidelines for improving the design of web applications. Web usage mining can assist in making a website adaptive by dynamically changing the content and structure of the site according to the patterns mined from user behavior.

- E-commerce.

  Discovering marketing intelligence from web usage data is critical for e-commerce applications. The knowledge acquired from the web usage mining about how customers are using a website facilitates effective Customer Relationship Management (CRM). Typically, in CRM, the main focus is on business specific issues such as customer attraction, customer retention, cross sales and customer departure.

2.2 Relational Fuzzy Subtractive Clustering (RFSC) Algorithm

The Relational Fuzzy Subtractive Clustering (RFSC) algorithm has been introduced by Suryavanshi et al. [59] based on the subtractive clustering algorithm [12]. The intention of RFSC algorithm is to group user-access log records into a number of classes for representing inherent fuzziness present in the user access behavior. The input to RFSC is a relation matrix R, showing pairwise dissimilarities among usage sessions. Each entry in R maintains a constraint of having normalized dissimilarity value, i.e., for K^{th} and L^{th} sessions, $0 \leq R_{KL} \leq 1$. Moreover, $R_{KL}=R_{LK}$ and $R_{KK}=0$.

RFSC algorithm starts by considering each session as a potential cluster center. It computes the potential of each session using its dissimilarity to the rest of the items in the dataset. Equation (1) presents the formula for computing the potential of a session us_K.

$$P_K = \sum_{L=1}^{N} e^{-\alpha R_{KL}^2} \tag{1}$$

Here, R_{KL} is the dissimilarity between usage sessions us_K and us_L, and N is the total number of sessions extracted from web log. And $\alpha = 4/\gamma^2$, where γ is the neighborhood-dissimilarity calculated from R with $0 \leq \gamma \leq 1$. Suryavanshi et al. [59] defined session-wise neighborhood-dissimilarity, denoted by γ_K, for each us_K as median of the dissimilarities of us_K to all other sessions, and the neighborhood-dissimilarity γ for the entire dataset as the median of all γ_K's.

After computing the potential of all sessions, the RFSC selects the session with the highest potential (P_1*) as first cluster center. Next, a subtraction step is performed, where the potentials of all sessions are reduced in proportion to their degree of similarity with the selected cluster center. Therefore, sessions possessing higher similarity to the selected cluster center face higher reduction in their potentials, leaving little chance to be selected for becoming the next cluster center. It should be noted that the reduction proportion is such that after the subtraction step, the potential of the selected cluster center is reduced to "zero".

After each subtraction, RFSC selects the next cluster center based on the modified potentials of sessions and two threshold values. The threshold values are called as the accept ratio, $\overline{\in}$ and the reject ratio, $\underline{\in}$ respectively, where $0 < \underline{\in}$, $\overline{\in} < 1$, and $\underline{\in} < \overline{\in}$. While selecting us_D with the highest modified potential as the next cluster center, it's potential (P_D) is compared with $\overline{\in}$ and $\underline{\in}$. If $P_D > \overline{\in} P_1*$, us_D is selected as the next cluster center. On the other hand, if $P_D < \underline{\in} P_1*$, us_D is rejected, and also terminates the RFSC algorithm. If $\underline{\in} P_1* < P_D < \overline{\in} P_1*$, us_D is selected as the next cluster center by checking whether it provides a good trade-off between having a sufficient potential and being sufficiently further from the existing cluster centers. Otherwise, we proceed with the next highest potential after making potential P_D to "zero". (For further details of the RFSC algorithm, see [59]).

At the end of RFSC algorithm, we obtain a set of Q clusters $C=\{C_1, C_2,...,C_Q\}$, where

each cluster center C_Z is an actual usage session us_Z, known as cluster prototype. The membership value of session us_D with each cluster C_Z is computed using equation (2).

$$MV_{C_Z D} = e^{-\alpha R^2_{C_Z D}} \tag{2}$$

Here, $Z \in [1...Q]$ and $D \in [1...N]$. $R_{C_Z D}$ is the dissimilarity between cluster prototype us_Z and usage session us_D. Sessions that are close to cluster prototypes have high membership values as compared to farther ones.

In our work, we use the RFSC algorithm for grouping sessions based on their dissimilarity. An important point to note is that a user may visit the site more than once with individual goals. Moreover, multiple accesses to the same page in the same session or in different sessions may be due to different sub-goals. Therefore, it is reasonable to represent the user browsing behavior through a number of fuzzy clusters, where each user is a member of every cluster with different membership value. As RFSC utilizes fuzzy techniques for web data clustering, it can better represent the fuzziness inherent in user browsing behavior by giving a set of fuzzy clusters. Besides, RFSC yields fairly accurate results, is scalable to very large datasets, is reasonably immune to noise present in web data, and is parameter independent. We have chosen to employ the RFSC algorithm in clustering of sessions taking these useful properties into account.

2.3 Web Recommender Systems and Web Usage Mining

The essence of web personalization is the adaptability of the site to the needs and interests of individual users. Typically, a personalized website recognizes user preferences and needs, and adapts it's services to assist the user in getting quickly to the information that he/she is seeking in the site. Thus, web personalization reduces to an extent the information overload problem that the users of a large website would usually face, and at the same time helps to satisfy the main goal of any website, which is the creation of loyal users. In general, web personalization could be based on one or more of the following types of web data [56].

- *Content:* Real data in web pages. This can be simple text, images, or structured data, such as information retrieved from databases.
- *Structure:* Organization of the content. The content can be HTML or XML tags within a page, or even hyperlinks connecting the pages to one another.
- *Usage:* Description of the usage pattern of web pages in a website, such as user IP addresses, accessed page references, date and time of accesses, etc.

Web recommender systems, one of the approaches used for web personalization, aim to recommend a list of hyperlinks to the user that are deemed to be the user's preference. These systems are implemented on the web server, and rely on the data showing user interest implicitly (i.e., browsing history as stored in server logs), or explicitly (i.e., explicit user ratings) [39]. Typically, recommender systems are used to make either a prediction about an item that a particular user is likely to prefer (prediction problem), or a recommendation of a set of items that will be of interest to a certain user (top-N recommendation problem). As mentioned in the previous chapter, there are three basic approaches, termed as filtering, used in the recommender systems for automatically providing recommendations to a user [15,39]: content-based filtering, manual rule-based filtering, and collaborative filtering. These are described further below:

2.3.1 Content-Based Filtering

Content-based filtering is based only on the interests/preferences of individual users. It recommends items to a user that are similar to the ones preferred by the user in the past. In particular, various candidate items are compared with items previously liked by the user, and the best-matching items are recommended. Item similarity is measured based on domain specific item attributes, such as, author and subject for book items, artist and genre for music items. Example of a system adopting this filtering techniques is NewsWeeder [25]. However, the content-based filtering faces problems due to limited features associated with the items, and finding all required features of each item is expensive.

2.3.2 Manual Rule-Based Filtering

In manual rule-based filtering, a set of rules are generated based on user demographics (i.e., general characteristics of users) or session history. Typically, these rules are used to recommend items to a particular user. In general, a user is asked to answer a set of online questions, until a customized result such as a list of products is obtained. Example of a system adopting the rule-based filtering approach is Yahoo!'s personalization engine [29]. However, this approach mostly depends on heavy planning, and the quality of the questions, possible answer combinations, and customizations by an expert. It also suffers from a lack of intelligence as there is no automatic learning, making it relatively static.

2.3.3 Collaborative Filtering

Collaborative filtering (CF) is the most popular and widely used approach to make recommendations about items such as web pages, movies, books, and etc. Many of the successful commercial systems such as "http://www.amazon.com" [27] and "www.CDNow.com" [22] are based on collaborative filtering approaches. The goal of collaborative filtering is to predict the preferences of a user, called as the current/active user, based on the preferences of a group of users. In other words, a collaborative filtering is a function that takes all past users' sessions as input, and provides recommendations for the pages that are not yet accessed by the active user [43]. Generally, it depends on the fundamental assumption that an active user will be interested in those items that are interesting to like-minded users. There are two major classes of collaborative filtering [7, 48] techniques: memory-based collaborative filtering and model-based collaborative filtering. Some have mentioned another class of collaborative filtering, which is called as hybrid collaborative filtering [43,68]. These are described below.

(a) Memory-Based Collaborative Filtering

In memory-based collaborative filtering, a database of all users' ratings of items is maintained in memory. A subset of users, called neighbors, are selected in real-time from this

database based on their similarity to the active user. The preferences of neighbors are combined to produce a prediction or top-N recommendations for the active user. Generally, a weighted average of deviations from the neighbors mean is used to make the prediction or recommendation. The memory-based CF is simple and easy to implement. However, it is hard to maintain prediction performance and accuracy using this approach due to the increasing sparsity in rating of items and also the increasing computation costs (i.e., time and memory requirements) of user similarity calculation and searching in real time as larger number of items and users are encountered [36].

(b) Model-Based Collaborative Filtering

In model-based collaborative filtering, a descriptive model of users/usage is built by applying various mining algorithms on all past user preferences. This model is later utilized to make the prediction or recommendation for the active user. As compared to memory-based, the model-based approach improves the scalability of collaborative filtering when dealing with larger datasets. Moreover, the recommendation time and memory requirement is minimized as compared to traditional collaborative filtering, since one is only considering the behavioral model instead of the whole database. However, the time complexity of required offline compilation of the data into a model may be expensive, and adding data about new items may require a full recompilation. In general, the performance of model-based collaborative filtering typically depends on the underlying modeling technique. In literature, there exists a number of model-based collaborative approaches used in the recommender systems, for details please see [2].

(c) Hybrid Collaborative Filtering

The hybrid CF approaches combine CF techniques with other recommender approaches, such as content-based filtering to make predictions or recommendations. In some research, the memory-based and the model-based CF approaches are combined to form hybrid CF approach.

We will discuss more about collaborative filtering systems in Chapter 5.

In recent years, there has been increasing interest in integration of web usage mining in web recommender systems [36]. Web usage mining is used to discover interesting usage patterns of the users from web log, and eventually generates an access behavioral model as a set of usage profiles (or aggregate usage profiles [36]) from the discovered patterns. Typically, each profile represents a weighted collection of pages that are frequently accessed by a group of users with common needs or interests. It is important to mention that web log stores all user browsing history, which contains useful information about users, and their interests and preferences. Therefore, web log data, which is representing user interests in an implicit manner, can be a good alternative in generating usage model in the absence of explicit user ratings of pages. In addition, clustering, a web usage mining technique, is a natural way to group similar items based on common properties. Therefore, by applying clustering to preprocessed web log records, it is possible to generate a set of useful usage profiles, where each profile gives an aggregate representation of the common interests of a group of users [36]. These profiles are helpful in better understanding the browsing behavior of users. Typically, each usage profile is represented as a set of page-weight pairs [36]:

$$pf_c = \{<url_i, \text{weight}(url_i, pf_c)> \mid \text{i}\in\text{M, c}\in\text{Q, and weight}(url_i, pf_c) \geq \mu\}$$

where weight(url_i,pf_c) is the popularity of url_i in the profile pf_c and μ is a threshold value used to prune out very low popular pages from the profile.

Fig. 2 shows a general framework for the personalization based on the web usage mining (also known as usage-based recommender systems). The overall recommendation task is divided into two classes: offline tasks and online tasks. The offline tasks involve the data preparation and the usage mining in order to generate a set of usage profiles. On the other hand, an online recommender engine receives these profiles as input, and utilizes them to produce recommendations in real time for the active user. The recommended items are added to the most recently requested page of the active user before sending it to him.

An important point to mention here is that the performance of usage-based recommender

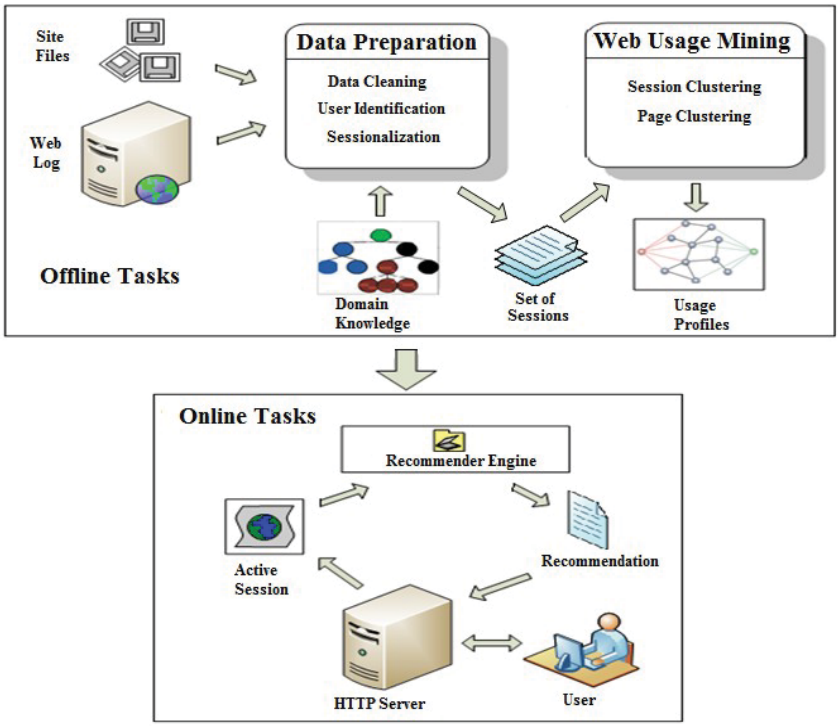


Figure 2: A General Framework of WUM-based Personalization System [36].

systems largely depends on the effectiveness of the access behavior model, i.e., the set of usage profiles obtained through web usage mining. Successful clustering of users is the key to effective usage profile generation. However, a good clustering depends on the accuracy of similarity computation among users. Again, accuracy of the user similarity computation depends on how user interests are captured. In this context, lot of research has been done to measure user interest, calculate user similarity by matching their browsing interests with various similarity measures, and eventually generate a set of usage profiles by grouping the users with similar interests. Most of these propose the use of web usage features such as page access frequency, page visiting time, and access sequence for estimating the user interest in a page [10,17,28,67]. We will discuss in detail these proposals in the next section.

2.4 Similarity Measures

Establishing the right similarity metric so as to capture the browsing interests of the user is crucial for grouping users. Efficient user grouping results in a set of effective usage profiles, which in turn enhances the performance of a usage-based recommender system by providing high quality recommendations to the users. One way is for the user to give a numeric rating to a page to show his/her interest [14]. Generally, a high rating for an item indicates a strong interest of user on it, whereas low value shows less interesting item. In another, user interests can be inferred by observing user access behavior from the web log, such as time spent on pages and/or page-visit frequencies [10, 17, 28, 67] or access sequence [11]. Typically, two users are said to be similar and should be in the same cluster, if they possess similar browsing interests. Considerable research has been conducted to establish methods which compute similarity among users based on their browsing data. Some give importance to the number of similar pages and their visiting order, while others pay attention to access frequency and/or duration [17].

One of the most popular measures employed to this aim is the cosine measure. In general, the cosine similarity between any two behavior vectors us_a and us_b can be computed using equation (3).

$$Cosine(us_a, us_b) = \frac{\sum_{j=1}^{m} us_{aj} us_{bj}}{\sqrt{\sum_{j=1}^{m} us_{aj}^2} \sqrt{\sum_{j=1}^{m} us_{bj}^2}} \tag{3}$$

In particular, Xia et al. [64, 65] proposed various versions of cosine similarity measures in order to capture similarity among user interests. Basically, they considered separate cosine similarity measure of the total number of common pages, their access frequency, their viewing time, and lastly their access order for similarity computation. They called these similarity measures as usage-based measure, frequency-based measure, viewing-time-based measure, and visiting-order-based measure respectively. Then they introduced a matrix-based clustering algorithm [65], and later, a multilevel clustering algorithm [64] for grouping users based on their similar interests. Finally, they integrated the resulting clusters

into a web document pre-fetching application. Shahabi et al. [51] created user profiles by capturing user selected links and took into consideration order of pages, viewing time, and cache references, using a JAVA remote agent. They computed similarity among navigation paths of the users using Cosine similarity, and grouped the users based on a path-mining algorithm. Later, Martin-Bautista et al. [30] presented a model for user profile generation by applying fuzzy C-means algorithm to the result of cosine similarity between two user sessions, where viewing time is used to represent user interest on the page. Some researchers preferred to use the Pearson correlation coefficient to measure correlation among user ratings, when explicit user rating on pages are available. The Pearson correlation between two vectors of user ratings R_a and R_b can be computed using equation (4), where $\overline{R}_a$ and $\overline{R}_b$ are the average rating of R_a and R_b respectively.

$$Pearson\_correlation(R_a, R_b) = \frac{\sum_{j=1}^{m}(R_{aj} - \overline{R}_a)(R_{bj} - \overline{R}_b)}{\sqrt{\sum_{j=1}^{m}(R_{aj} - \overline{R}_a)^2}\sqrt{\sum_{j=1}^{m}(R_{bj} - \overline{R}_b)^2}} \tag{4}$$

In this context, Xue et al. [66] proposed a novel approach of combining memory-based and model-based collaborative filtering by introducing a cluster-based smoothing method. They generated a set of clusters using the K-means algorithm, with similarity among user ratings measured by the Pearson correlation. They utilized these clusters for smoothing unrated items of individual users, and also for selecting the neighborhood to make recommendations. Following the same similarity equation for user ratings, Keqin et al. [24] proposed a new collaborative filtering algorithm based on user interest partitioning. They divided the user interests (represented by ratings) into pieces, called as an interest unit, and computed the similarity between users on interest unit, referred to as local similarity. They also measured the user similarity based on the whole interests, called as holistic similarity. Note that, both of these similarity computations involved Pearson correlation. These two similarity results are linearly combined for searching the nearest neighbors and for generating recommendations. Sarwar et al. [48] analyzed different item-based recommendation generation algorithms in order to provide high quality recommendations for large-scale dataset.

They computed item to item similarity by applying the Pearson correlation and the cosine measure between item-rating vectors. They used a weighted sum and regression model for providing recommendations. Similarly, Breese et al. [7] applied the Pearson correlation and the cosine measure for similarity weighting, and performed an empirical analysis of several variants of neighborhood-based collaborative filtering approaches.

Besides the Pearson correlation and the cosine similarity, some research works involve the Jaccard coefficient measure for computing similarity among sessions. In general, the Jaccard coefficient between any two behavior vectors us_a and us_b can be computed using equation (5).

$$Jaccard(us_a, us_b) = \frac{|us_a \cap us_b|}{|us_a \cup us_b|} \tag{5}$$

Considering Jaccard coefficient, Nadi et al. [38] proposed a hybrid recommender system by combining collaborative and content-based filtering approaches. They used Jaccard coefficient for measuring similarity among documents. They utilized the computed similarity to group the documents into a set of document clusters. They represented each user access behavior with respect to the document clusters (called access matrix), where user interest on a particular document cluster was measured by taking the total number of accesses to the document cluster normalized by the total number of accesses to all document clusters. They applied a two phase clustering algorithm by combining the ant-based algorithm and the fuzzy C-means algorithm on the access matrix to group users. They used the clusters to provide recommendations to an active user. Santhisree et al. [47] proposed a new sequence similarity measure (called as SSM) by combining Jaccard similarity of two session sequences, the frequency count of pages accessed in these sequences, and the total length of sub-sequence common to them. They considered all accessed pages in the session sequence to be equally interesting to the user. They introduced two clustering techniques, named as SSM-density based clustering algorithm and SSM-optics based clustering algorithm for finding meaningful clusters of users.

Considering the presence of uncertainty and fuzziness in user browsing behavior, in a lot

of research on this topic, fuzzy sets are used to present the user interests on pages. In this context, Castellan et al. [8] used access-time to show user interest on a page, and generated a fuzzy set of access-time by utilizing two time thresholds t_{min} and t_{min}. After measuring user similarity using their proposed fuzzy Jaccard coefficient, they applied the CARD+ (Competitive Agglomeration Relation Data) algorithm to generate user profiles (i.e., usage profiles). In a similar manner, Wang et al. [63] developed a fuzzy multiset to characterize user interests by integrating page-click rate, viewing-time, and user's preference, and applied the traditional max-min approach to generate multi-fuzzy similarity matrix, representing user browsing similarity. They proposed CAFM (Clustering Algorithm based on Fuzzy Multisets) algorithm to group pages and users as well. Applying the same max-min measure, Yu et al. [69] later generated a fuzzy similarity matrix based on similarity in user interests, where they took into account page-click number and web browsing time as indicators of user interest. Finally, they grouped similar users using a novel fuzzy clustering method. An important point to note is that all of these similarity computations are based only on common pages between the two usage sessions being compared.

It has been found that inclusion of website structural information or prior domain knowledge with web usage data provides better quality in user similarity [6, 34] computations. Typically, pages in a website are organized according to a hierarchical relationship based on their subject (topic) similarity. [40] quantified this relationship among page URLs as a distance measure and incorporated it into session similarity measure for clustering users. They considered each usage session as a binary vector with all accessed pages having equal degree of user interest. Later, [26] incorporated page similarity, computed from URL-similarity and viewing-time similarity, in a sequence alignment method for measuring user similarity.

From the above, it is clear that considerable amount of research has focused on finding similarity between usage sessions. Basically, there are two somewhat distinct directions - use of website structure in computing similarity and incorporation of user interests when computing similarity. There is not much work on suitably combining the two so as to benefit from both aspects. In the following chapters, we define a new weighted similarity

measure by integrating user's interests in pages and URL-structure similarity of pages. We make use of web log for generating a set of weighted sessions by capturing user interests on pages. Using this weighted similarity measure, we compute similarity among weighted sessions. Following that, we generate a set of fuzzy clusters of the sessions by applying the RFSC algorithm to the computed similarity. Subsequent processing of clusters leads to a set of usage profiles. Finally, these profiles are utilized by recommender algorithms to make recommendations to the user.

Chapter 3

Preprocessing - Creation of Weighted Sessions

This chapter presents our methods for measuring degree of user interest in the web pages while browsing and incorporating them into the format of a weighted session. In order to set the stage, we first discuss the dataflow diagram of our recommender systems. Next, we explain the process of extracting usage sessions from the web log records. Then, we estimate the session-wise page significance to measure degree of user interests. Finally, we convert the usage sessions into weighted sessions by considering page-significance.

3.1 Recommender System Architecture

Generally, all activities related to the web personalization based on usage profiles can be divided into two separate classes: offline activities and online activities [36]. The offline activities deal with the web data preparation and the usage profiles generation, whereas the online activities are responsible for recommending pages to an active user. Fig. 3 shows the dataflow diagram of our WUM-based personalization system architecture. There are three major modules:

- Web Log Preprocessing (WLP)

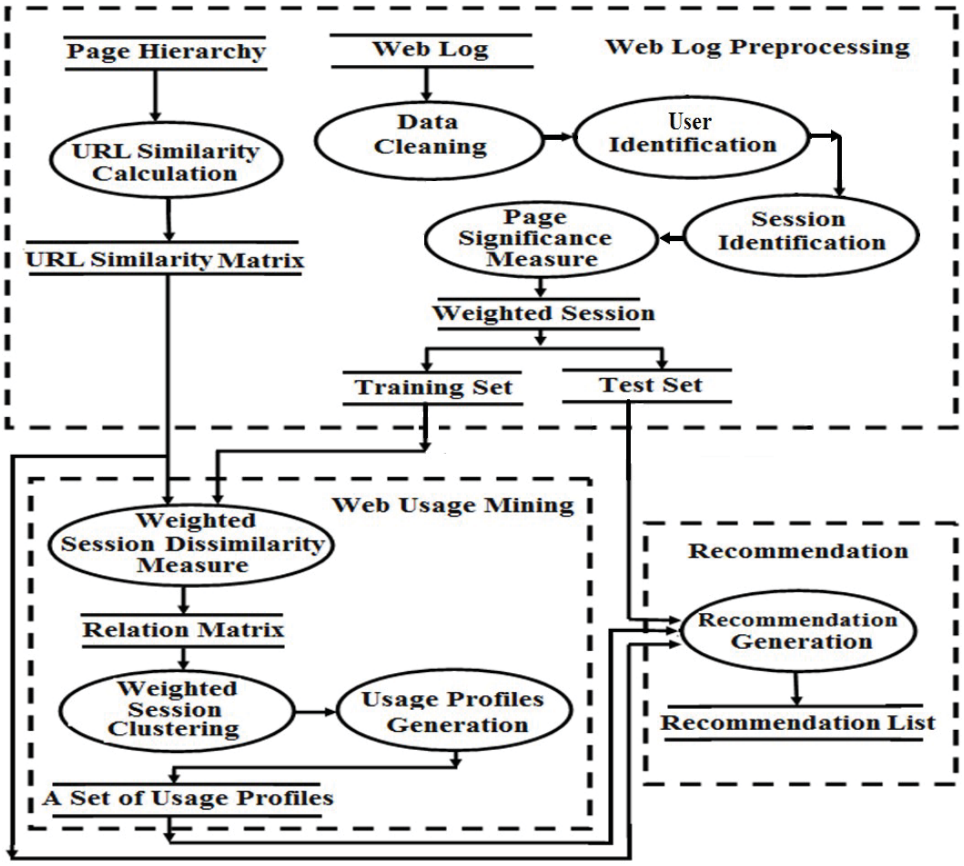


Figure 3: The Dataflow Diagram of Our Personalization System.

- Web Usage Mining (WUM)
- Recommendation

The WLP module deals with session extraction, page-significance computation and creation of weighted sessions. A set of usage sessions are extracted from Web log records by applying data preprocessing method. Each usage session consists of a sequence of accessed pages, together with their access frequency and duration. Utilizing these access frequency and duration, session-wise page significance is measured. The WUM module handles weighted session similarity computation and usage profiles generation. Applying session-wise page significance, the extracted usage sessions are converted into weighted sessions. For experiments used in evaluation of the proposed methods, these weighted sessions are partitioned into a training set and a test set.

The test set is retained for use in generation of online recommendations during the testing phase, and the training weighted sessions are used for offline generation of usage profiles. The similarity among training weighted sessions is determined by utilizing page significance and their URL-structure similarity. A set of usage profiles is generated by applying the RFSC clustering method [59] on the session similarity results. Finally, performance of the recommendation module will be tested using part of the pages in each of the weighted sessions in the test set and recommending the other pages. Recommendation is based on the usage profiles.

3.2 Usage Session Extraction

A web server logs all user browsing activities as a sequence of records. From the given web log, we first find basic usage information, which includes user IP address, requested date and time, requested URL, HTTP status code, file-size in bytes. After cleaning irrelevant entries such as requests from web robots or crawlars, any image file entries or requests with unsuccessful HTTP status code, we use the IP address to group requests of individual user. This is under the assumption, that a sequence of web activities from the same IP address are very likely to be from the same user (anonymous). And it should be noted that this is the practice in processing web log data. We apply two time-oriented heuristics for session extraction [4, 5, 9, 28, 54, 70]. They are the session-duration heuristic and the page-stay time heuristic. We use 30 minutes as a threshold (θ) for session duration, and also 10 minutes as a threshold (β) for page-stay time. A new session is considered when either θ or β is exceeded.

Let U=$\{\text{url}_1,\text{url}_2,...,\text{url}_M\}$ be a set of M pages in the website under study. Let US=$\{\text{us}_1, \text{us}_2,...,\text{us}_N\}$ be the set of N usage sessions extracted from the web log. Each usage session comprises of a sequence of a subset of U together with access duration, frequency, and size. It is presented as $\text{us}_K = \{(\text{url}_1,\text{t}_1,\text{f}_1,\text{size}_1), (\text{url}_2,\text{t}_2,\text{f}_2,\text{size}_2), ..., (\text{url}_M, \text{t}_M,\text{f}_M,\text{size}_M)\}$, where url_j, t_j, f_j, and size_j are the visited page, access time in seconds, access frequency and size (# of bytes) respectively.

3.3 Degree of User Interests

A web user may visit a website more than once with individual goals. While browsing each time, the user finds some of the pages more interesting, and the other pages less. The latter may be visited for other reasons like navigation, accidental visit, casual exploration, etc. This implies that accessed pages are of interest to a user with varying degrees. Therefore, a page-significance measure could be used for estimating the user interest in the page. It should be noted that significance of a page in a session indicates the degree of user interest in the page in that specific session. It is reasonable to assume that access frequency and duration are two major indicators of a user interest in a page [10, 28]. Inspired by this we propose a page-significance weighted measure for estimating user interest, as described next.

Page access duration does indicate degree of user interest, but it must also depend on the content in the page. Generally, when a user spends longer time on a page, it is likely that the page is of interest to him/her. However, a quick move to another page might be due to the small content in that page (size in bytes). Therefore, user interest in a page in a session by means of "duration" can be estimated by the time spent on a page with respect to its size. This is further normalized by the maximum in the session to recognize the importance of that page compared to other pages. A point to note is that we consider cumulative time duration on a page due to multiple access to the page in a session. Equation (6) estimates user interest in a page url_j in a session us_K as regards "access duration", where $0 \leq \text{Duration}_{url_j} \leq 1$.

$$Duration_{url_j} = \frac{\frac{\sum access\_time_j}{page\_size_j}}{max(\forall_{r \in us_K} \frac{\sum access\_time_r}{page\_size_r})} \tag{6}$$

Again, a user may go back to visit an interesting page in a single session. Hence, user interest associated with a page in a session using "frequency" can be measured by the number of visits normalized by the maximum number of visits in that session. Equation (7)

measures user interest in a page url_j in a session us_K as regards "access frequency", where $0 \leq \text{Frequency}_{url_j} \leq 1$.

$$Frequency_{url_j} = \frac{\sum visit_j}{max(\forall_{r \in us_K} \sum visit_r)} \tag{7}$$

Paying equal importance to all the factors mentioned above, our page-significance measure utilizes a harmonic mean of Duration_{url_j} and Frequency_{url_j} for estimating user interest in a page. We use harmonic mean since it tends to mitigate the impact of large outliers and aggravate the impact of small ones. Equation (8) shows the formula of "page significance" for a page url_j in a session us_K, where $0 \leq \text{Sig}_{url_j} \leq 1$.

$$Sig_{url_j} = \frac{2 \times Duration_{url_j} \times Frequency_{url_j}}{Duration_{url_j} + Frequency_{url_j}} \tag{8}$$

Equation (8) confirms that the interest of a user on a page in a session is high when both of access duration and access frequency are high.

3.4 Weighted Usage Session Conversion

Let, N denote the number of usage sessions extracted from the given web log. For generating weighted sessions, we measure session-wise significance of all pages using equations (6) to (8). Typically, more significance to a page means more interesting page to the user. Further, the most significant page is given rank 1 and the remaining pages are ranked accordingly. We term each session with a set of accessed pages together with their significance and rank as *Weighted Session*. It is represented as $\text{ws}_K = \{(\text{url}_1, \text{Sig}_{url_1}, \text{rk}_1), (\text{url}_2, \text{Sig}_{url_2}, \text{rk}_2), ..., (\text{url}_M, \text{Sig}_{url_M}, \text{rk}_M)\}$, where url_j, Sig_{url_j}, and rk_j are the visited page, its significance weight, and rank respectively. Below, the weighted session conversion process is explained using an example.

Example 1.

Let $\text{us}_r = \{(\text{url}_1,150,2,2500), (\text{url}_2,100,1,4000), (\text{url}_3,100,2,1000), (\text{url}_5,150,1,5000)\}$ be an extracted usage session. By using formulas 6 to 8, we find Sig_{url_1}=0.75, Sig_{url_2}=0.33,

Sig_{url_3}=1.0, and Sig_{url_5}=0.38. Next, by ranking the pages based on their significance, we get weighted session ws_r = {(url_1,0.75,2), (url_2,0.33,4), (url_3,1.0,1), (url_5,0.38,3)}.

The next chapter describes how this weighted session is used to derive a new usage session similarity measure which captures user browsing interests in addition to URL-structure similarity.

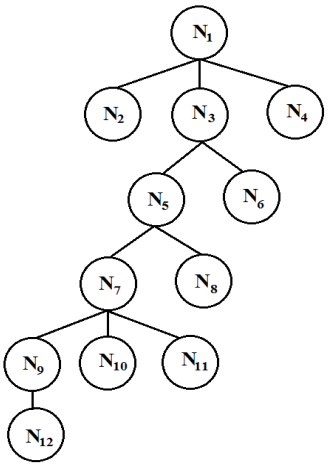
Chapter 4

Weighted Session Similarity Measure

In this chapter, we first propose a modification to URL structure-based similarity measure for pages (URL similarity, for short) so as to accommodate the fact that pages deeper in the hierarchy are more specialized in content. Then we present our formulation for a weighted session similarity measure which includes both URL similarity and page significance.

4.1 Web Directory Structure

The URL structure of websites is hierarchical. The intention is to assist users to narrow down into a topic. Each non-leaf node belongs to a page-URL corresponding to a directory (i.e., /folder_name/) of the web server. Each leaf represents a page-URL that corresponds to a file (i.e., /folder_name/file_name.html). The root corresponds to the URL of home page of the website. In our work, we consider the root at level "one" of the hierarchy, L_1. Any internal node at level L_k is directly linked to all of its children in next lower level L_{k+1} via individual edges. This may be assumed to imply a "Consists-of" relationship between them. Usually, the web pages sharing similar subject are more structurally related, and their positions in the hierarchy are influenced by their URLs. Therefore, computing the similarity among pages via their URLs is one way of capturing topic-based similarity



Node	URLS	Level No.
N_1	http://www.cs.xxx.yy	1
N_2	http://www.cs.xxx.yy/program/	2
N_3	http://www.cs.xxx.yy/people/	2
N_4	http://www.cs.xxx.yy/research/	2
N_5	http://www.cs.xxx.yy/people/faculty/	3
N_6	http://www.cs.xxx.yy/people/staff/	3
N_7	http://www.cs.xxx.yy/people/faculty/prof1/	4
N_8	http://www.cs.xxx.yy/people/faculty/prof2/	4
N_9	http://www.cs.xxx.yy/people/faculty/prof1/course/	5
N_{10}	http://www.cs.xxx.yy/people/faculty/prof1/profile.html	5
N_{11}	http://www.cs.xxx.yy/people/faculty/prof1/publications.html	5
N_{12}	http://www.cs.xxx.yy/people/faculty/prof1/course/lecture_notes.html	6

Figure 4: A Part of Page Hierarchy of a "University CS Department" Website.

among users accessing those pages. As an example for discussion, Fig. 4 shows a part of page hierarchy of a "University CS Department" website.

4.2 URL Similarity Measure

Generally, web pages sharing similar topics are structurally related by URLs. Looking at the paths leading to the pages from the root, it is possible to discover similarity among pages. This is useful in capturing subject similarity in user interests. Consider the following pair of URLs, url_1="http://www.cs.xxx.yy/people/faculty/prof1/profile.html" and url_2="http://www.cs.xxx.yy/people/faculty/prof1/publications.html" (using Fig. 4). Both of these pages convey information about a particular professor's profile and his publications. Therefore, similarity between url_1 and url_2 is obvious. Again, consider another pair of URLs, url_1="http://www.cs.xxx.yy/people/faculty/prof1/profile.html" and url_3="http://www.cs.xxx.yy/people/faculty/prof2/" (using Fig.4). The latter pair should be less similar as compared to the first pair, as they convey information of two different faculty members. Therefore, a systematic approach is needed to numerically compute the similarity between pages through their URLs, signifying the fact that while going down through a path in page hierarchy, topics are becoming more specialized and nodes are more conceptually related.

In this context, Nasraoui et al. [40] defined a URL similarity measure for pages with the consideration that larger overlap in URLs must result in a higher similarity between pages. They used a URL similarity of "one" for any node and its parent, and also for sibling nodes sharing the same parent. Their URL similarity measure between two urls url_i and url_j is shown in equation (9), where $0 \leq \text{S}_u(\text{i,j}) \leq 1$.

$$S_u(i,j) = min(1, \frac{|url_i \cap url_j|}{max(1, max(|url_i|, |url_j|) - 1)}) \tag{9}$$

However, we believe that it is better to not assume a similarity of "one" for the pairs located at different levels of a hierarchy. In fact, more specialized information is likely to be derived from lower-level nodes compared to upper-level nodes. Moreover, as one dips more into a hierarchy, the topics in the URLs are conceptually more related. Therefore, we argue that any sibling/parent-child URL pairs positioned at deeper level(s) should possess greater similarity than those pairs at upper level(s).

Considering this, we define a URL similarity measure among pages based on their positions in a page hierarchy. The proposed URL similarity has three important features. Firstly, any two URLs url_i and url_j with more overlap in the hierarchy possess higher similarity than any other pairs with lesser overlap. Secondly, any pair of sibling URLs at L_n has higher similarity than any sibling pair at L_k when k<n. Lastly, any URL at L_n and it's parent URL at level L_{n-1} is more similar than any URL at L_k and it's parent at level L_{k-1}, when k<n. Our proposed URL similarity for url_i and url_j is defined in equation (10).

$$URL_{sim}(i,j) = \frac{L(url_i \cap url_j)}{max(L(url_i), L(url_j))} \tag{10}$$

Here $\text{L}(\text{url}_i)$ is the level number of a node N_i related to url_i in the hierarchy and $\text{L}(\text{url}_i \cap \text{url}_j)$ is the level of common ancestor node url_i and url_j. Our URL similarity measure satisfies the following properties:

1. $\text{URL}_{sim}(\text{url}_i,\text{url}_j) = \text{URL}_{sim}(\text{url}_j,\text{url}_i)$
2. $0 < \text{URL}_{sim}(\text{url}_i,\text{url}_j) \leq 1$

The following two examples illustrate how this proposed URL similarity measure can better represent the page similarity for a university website and a commercial website ("www.rbcroyalbank.com").

Table 3: URL similarity matrix for Fig. 4 using equation 9 [40].

Node	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}
N_1	1.0	1.0	1.0	1.0	0.50	0.50	0.33	0.33	0.25	0.25	0.25	0.20
N_2	1.0	1.0	1.0	1.0	0.50	0.50	0.33	0.33	0.25	0.25	0.25	0.20
N_3	1.0	1.0	1.0	1.0	1.0	1.0	0.67	0.67	0.50	0.50	0.50	0.40
N_4	1.0	1.0	1.0	1.0	0.50	0.50	0.33	0.33	0.25	0.25	0.25	0.20
N_5	0.50	0.50	1.0	0.50	1.0	1.0	1.0	1.0	0.75	0.75	0.75	0.60
N_6	0.50	0.50	1.0	0.50	1.0	1.0	0.67	0.67	0.50	0.50	0.50	0.40
N_7	0.33	0.33	0.67	0.33	1.0	0.67	1.0	1.0	1.0	1.0	1.0	0.80
N_8	0.33	0.33	0.67	0.33	1.0	0.67	1.0	1.0	0.75	0.75	0.75	0.60
N_9	0.25	0.25	0.50	0.25	0.75	0.50	1.0	0.75	1.0	1.0	1.0	1.0
N_{10}	0.25	0.25	0.50	0.25	0.75	0.50	1.0	0.75	1.0	1.0	1.0	0.80
N_{11}	0.25	0.25	0.50	0.25	0.75	0.50	1.0	0.75	1.0	1.0	1.0	0.80
N_{12}	0.20	0.20	0.40	0.20	0.60	0.40	0.80	0.60	1.0	0.80	0.80	1.0

Example 2.

Let us consider the page hierarchy shown in Fig. 4. Table 3 presents the URL similarity matrix using equation (9) proposed by Nasraoui et al. [40] (with respect to Fig. 4). From Table 3, N_3="http://www.cs.xxx.yy/people/" and N_9="http://www.cs.xxx.yy/people/faculty/prof1/course/" has a URL similarity of 0.50, whereas for N_7="http://www.cs.xxx.yy/ people/faculty/prof1/" and N_{12}="http://www.cs.xxx.yy/people/faculty/prof1/course/lecture_notes.html" it is 0.75 which is obvious. Further, as already mentioned, they considered that URL similarity between a node at any level and its parent node is always "one". Therefore, N_3="http://www. cs.xxx.yy/people/" and N_5="http://www .cs.xxx.yy/people/faculty/", conveying general information of faculty members, possess a URL similarity of "one" which is identical to the similarity between N_9="http://www.cs.xxx.yy/people/faculty/prof1/course/" and N_{12}="http://www.cs.xxx.yy/people/faculty/prof1/course/lecture_notes.h-

tml", showing relatively more specific information about a professor's course. This property is retained for all other parent-child pairs according to their measure. In addition, they assigned a URL similarity of "one" for any two sibling nodes sharing the same parent. Therefore, for sibling N_2="http://www.cs.xxx.yy/program/" and N_4="http://www.cs.xxx.yy/research/" sharing general information about program (i.e., graduate, under-graduate, and etc.) and research activities, the URL similarity is "one". The same similarity of "one" is obtained from their measure for siblings N_9="http://www.cs.xxx.yy/people/faculty/prof1/course/" and N_{10}="http://www.cs.xxx.yy/people/faculty/prof1/profile.html", sharing information about a particular professor. This property is also kept for all other siblings.

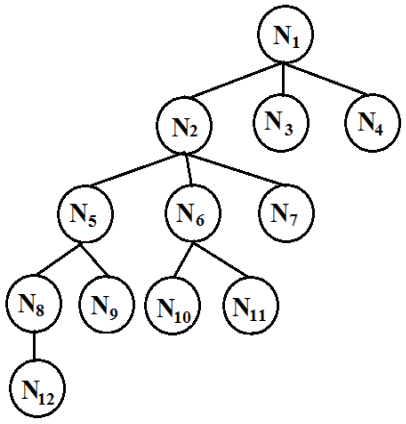
Table 4: URL similarity matrix for Fig. 4 using our proposed similarity measure.

Node	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}
N_1	1.0	0.50	0.50	0.50	0.33	0.33	0.25	0.25	0.20	0.20	0.20	0.17
N_2	0.50	1.0	0.50	0.50	0.33	0.33	0.25	0.25	0.20	0.20	0.20	0.17
N_3	0.50	0.50	1.0	0.50	0.67	0.67	0.50	0.50	0.40	0.40	0.40	0.33
N_4	0.50	0.50	0.50	1.0	0.33	0.33	0.25	0.25	0.20	0.20	0.20	0.17
N_5	0.33	0.67	0.67	0.33	1.0	0.67	0.75	0.75	0.60	0.60	0.60	0.50
N_6	0.33	0.67	0.67	0.33	0.67	1.0	0.50	0.50	0.40	0.40	0.40	0.33
N_7	0.25	0.50	0.50	0.25	0.75	0.50	1.0	0.75	0.80	0.80	0.80	0.67
N_8	0.25	0.50	0.50	0.25	0.75	0.50	0.75	1.0	0.60	0.60	0.60	0.50
N_9	0.20	0.40	0.40	0.20	0.60	0.40	0.60	0.60	1.0	0.80	0.80	0.83
N_{10}	0.20	0.40	0.40	0.20	0.60	0.40	0.60	0.60	0.80	1.0	0.80	0.67
N_{11}	0.20	0.40	0.40	0.20	0.60	0.40	0.60	0.60	0.80	0.80	1.0	0.67
N_{12}	0.17	0.33	0.33	0.17	0.50	0.33	0.50	0.67	0.83	0.67	0.67	1.0

Table 4 shows the URL similarity matrix (with respect to Fig.4) based on our proposed URL similarity measure using equation (10). From Table 4, we obtain $URL_{sim}(N_3,N_9)$=0.40 and $URL_{sim}(N_7,N_{12})$=0.67. This formulation yields similarity values such that URLs with larger overlap do show higher similarity. In addition, $URL_{sim}(N_2,N_4)$=0.50 and $URL_{sim}(N_9,N_{10})$=0.80. This shows that sibling URLs at deeper level have higher similarity than any sibling pair at upper level. Also, $URL_{sim}(N_3,N_5)$=0.67 and $URL_{sim}(N_9,N_{12})$=0.83, i.e., parent-child URLs at deeper level are more alike than a parent-child pair at upper level.

Example 3.

Fig. 5 represents a part of the page hierarchy of "RBC Royal Bank" of Canada website ("www.rbcroyalbank.com").



Node	URLS	Level No.
N_1	http://www.rbcroyalbank.com/	1
N_2	http://www.rbcroyalbank.com/personal-banking/	2
N_3	http://www.rbcroyalbank.com/business-banking/	2
N_4	http://www.rbcroyalbank.com/commercial-banking/	2
N_5	http://www.rbcroyalbank.com/personal-banking/product/	3
N_6	http://www.rbcroyalbank.com/personal-banking/credit-cards/	3
N_7	http://www.rbcroyalbank.com/personal-banking/ personal-loans /	3
N_8	http://www.rbcroyalbank.com/personal-banking/product/deposits/	4
N_9	http://www.rbcroyalbank.com/personal-banking/product/gic/	4
N_{10}	http://www.rbcroyalbank.com/personal-banking/credit-cards/reward-credit-cards/	4
N_{11}	http://www.rbcroyalbank.com/personal-banking/credit-cards/travel-credit-cards/	4
N_{12}	http://www.rbcroyalbank.com/personal-banking/product/deposits/saving-accounts.html	5

Figure 5: A Part of Page Hierarchy of "RBC Royal Bank" Website.

Table 5: URL similarity matrix for Fig. 5 using equation 9 [40].

Node	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}
N_1	1.0	1.0	1.0	1.0	0.50	0.50	0.50	0.33	0.33	0.33	0.33	0.25
N_2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.67	0.67	0.67	0.67	0.50
N_3	1.0	1.0	1.0	1.0	0.50	0.50	0.50	0.33	0.33	0.33	0.33	0.25
N_4	1.0	1.0	1.0	1.0	0.50	0.50	0.50	0.33	0.33	0.33	0.33	0.25
N_5	0.50	1.0	0.50	0.50	1.0	1.0	1.0	1.0	1.0	0.67	0.67	0.75
N_6	0.50	1.0	0.50	0.50	1.0	1.0	1.0	0.67	0.67	1.0	1.0	0.50
N_7	0.50	1.0	0.50	0.50	1.0	1.0	1.0	0.67	0.67	0.67	0.67	0.50
N_8	0.33	0.67	0.33	0.33	1.0	0.67	0.67	1.0	1.0	0.67	0.67	1.0
N_9	0.33	0.67	0.33	0.33	1.0	0.67	0.67	1.0	1.0	0.67	0.67	0.75
N_{10}	0.33	0.67	0.33	0.33	0.67	1.0	0.67	0.67	0.67	1.0	1.0	0.50
N_{11}	0.33	0.67	0.33	0.33	0.67	1.0	0.67	0.67	0.67	1.0	1.0	0.50
N_{12}	0.25	0.50	0.25	0.25	0.75	0.50	0.50	1.0	0.75	0.50	0.50	1.0

Table-5 represents the URL similarity matrix for Fig. 5 using equation (9) [40]. From Table 5, for N_5="http://www.rbcroyalbank.com/personal-banking/product/" and N_{10}="http://www.rbcroyalbank.com/personal-banking/credit-cards/reward-credit-cards/", we find $S_u(N_5,N_{10})$=0.67, and for N_5="http://www.rbcroyalbank.com/personal-banking/product/"

and N_{12}="http://www.rbcroyalbank.com/personal-banking/product/deposits/saving-accounts.html", we get $S_u(N_5,N_{12})$=0.75, i.e., more overlapping shows more similarity. Further, despite of sharing general information about personal bank accounts only, the parent-child pair N_2="http://www.rbcroyalbank.com/personal-banking/" and N_5="http://www.rbcroyalbank.com/personal-banking/product/" has similarity of "one". Again, N_6="http://www.rbcroyalbank.com/personal-banking/credit-cards/" corresponds to a page showing personal bank credit cards information, whereas N_7="http://www.rbcroyalbank.com/personal-banking/personal-loans/" corresponds to a page containing personal loan information (using Fig. 5). Therefore, these two pages convey different information. But they are siblings with common parent node N_2, therefore according to equation (9), $S_u(N_6,N_7)$=1.0. In a similar, $S_u(N_8,N_9)$=1.0 for sibling pair N_8="http://www.rbcroyalban- k.com/personal-banking/product/deposits/" and N_9="http://www.rbcroyalbank.com/personal-banking/product/gic/".

Table 6: URL similarity matrix for Fig. 5 using our proposed similarity measure.

Node	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}
N_1	1.0	0.50	0.50	0.50	0.33	0.33	0.33	0.25	0.25	0.25	0.25	0.20
N_2	0.50	1.0	0.50	0.50	0.67	0.67	0.67	0.50	0.50	0.50	0.50	0.40
N_3	0.50	0.50	1.0	0.50	0.33	0.33	0.33	0.25	0.25	0.25	0.25	0.20
N_4	0.50	0.50	0.50	1.0	0.33	0.33	0.33	0.25	0.25	0.25	0.25	0.20
N_5	0.33	0.67	0.33	0.33	1.0	0.67	0.67	0.75	0.75	0.50	0.50	0.60
N_6	0.33	0.67	0.33	0.33	0.67	1.0	0.67	0.50	0.50	0.75	0.75	0.40
N_7	0.33	0.67	0.33	0.25	0.67	0.67	1.0	0.50	0.50	0.50	0.50	0.40
N_8	0.25	0.50	0.25	0.25	0.75	0.50	0.50	1.0	0.75	0.50	0.50	0.80
N_9	0.25	0.50	0.25	0.25	0.75	0.50	0.50	0.75	1.0	0.50	0.50	0.80
N_{10}	0.25	0.50	0.25	0.25	0.50	0.75	0.50	0.50	0.50	1.0	0.75	0.40
N_{11}	0.25	0.50	0.25	0.25	0.50	0.75	0.50	0.50	0.50	0.75	1.0	0.40
N_{12}	0.20	0.40	0.20	0.20	0.60	0.40	0.40	0.80	0.80	0.40	0.40	1.0

Table 6 represents the URL similarity matrix for Fig. 5 based on proposed URL similarity measure using equation (10). From Table 6, $URL_{sim}(N_5,N_{10})$=0.50 and $URL_{sim}(N_5,N_{12})$ =0.60, i.e., URLs with greater overlapping provides larger similarity. Again, $URL_{sim}(N_6,N_7)$

=0.67 and $\text{URL}_{sim}(\text{N}_8, \text{N}_9)$=0.75, i.e., sibling pair at deeper level possesses higher similarity than any siblings at upper level. Also, $\text{URL}_{sim}(\text{N}_2,\text{N}_5)$=0.67 and $\text{URL}_{sim}(\text{N}_8,\text{N}_{12})$ =0.80, i.e., a parent-child URL pair at deeper level are more alike than a parent-child URL pair at upper level.

4.3 Weighted Session Similarity Measure

Let WS=$\{\text{ws}_1,\text{ws}_2,...,\text{ws}_N\}$ be a set of weighted sessions, where N is the total number of weighted sessions. Each weighted session ws_K is a set of visited URLs together with their significance and rank respectively, i.e., ws_K=$\{(\text{url}_1, \text{Sig}_{url_1}, \text{rk}_1), (\text{url}_2, \text{Sig}_{url_2}, \text{rk}_2), ..., (\text{url}_M, \text{Sig}_{url_M}, \text{rk}_M)\}$, where url_i, Sig_{url_i}, and rk_i are the visited page, its significance weight, and rank respectively with $1\leq\text{K}\leq\text{N}$ and $1\leq\text{i}\leq\text{M}$.

The proposed new similarity measure for weighted sessions incorporates page significance and their URL similarity. Let us recall that page significance indicates user interest on the page, and the URL similarity between pages shows the subject (topic) similarity between interests of any two users. The browsing interests of two users can be said to be comparable when they access similar pages with similar page significance and similar subjects. The weighted session similarity *WSS* is defined as the maximum of two other measures: cosine similarity, WSS_1 and structure-based cosine similarity, WSS_2 to measure similarity between weighted sessions. The WSS_1 determines cosine similarity between sessions, based on the significance of identical pages and completely ignores structural relation of pages. According to WSS_1, a similarity of "one" is assigned for identical sessions with equal page significance, but similarity score may vary with difference in the significance. A similarity score of "zero" is assigned when the pages are different, independent of their positions in the page hierarchy. Equation (11) shows WSS_1 measure for two weighted sessions ws_K and ws_L, where $0\leq WSS_1\leq 1$.

$$WSS_{1KL} = \frac{\sum_{i=1}^{M} ws_K(Sig_{url_i}) \times ws_L(Sig_{url_i})}{\sqrt{\sum_{i=1}^{M} ws_K(Sig_{url_i})^2}\sqrt{\sum_{j=1}^{M} ws_L(Sig_{url_j})^2}} \tag{11}$$

The following examples illustrate the effect of this equation, using the page hierarchy shown in Fig. 4. For example, consider the following two weighted sessions ws_P={(N_7,0.811, 1),(N_{10},0.756,2)} and ws_Q={(N_7,0.291,2),(N_{10},0.619,1)}. By this equation, they both are assigned the similarity score of 0.928. The pair has identical pages, but different significance, showing different interests of users.

Consider another example, WSS_1 for session pair ws_K={(N_7,0.672,1),(N_9,0.521,2)} and ws_L={(N_{10},0.431,2),(N_{11},0.542,1)}. They both are assigned the similarity value of 0 due to all pages being different. In a similar manner, consider ws_K={(N_7,0.672,1),(N_9,0.521,2)} and ws_R={(N_3,0.811,1),(N_4,0.289,2)}. These also are assigned the similarity value of 0. But, if we observe the URLs more carefully, we can easily see that ws_K is actually more similar to ws_L than ws_R, if we take into consideration structural similarity among pages (see Fig. 4). In fact, ws_K and ws_L both appear to be interested in a particular professor's profile (see Fig. 4), whereas it is difficult to presume this for the pair ws_K and ws_R. Therefore, WSS_1 clearly has some limitations in adequately representing such structural similarity among sessions. In contrast, WSS_2 is defined so as to overcome this limitation. It incorporates both URL similarity and page significance. Equation (12) provides the formulation of WSS_2 for ws_K and ws_L, where $0 < WSS_2 \leq 1$.

$$WSS_{2KL} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{M} ws_K(Sig_{url_i}) \times ws_L(Sig_{url_j}) \times URL_{sim}(i,j)}{\sum_{i=1}^{M} ws_K(Sig_{url_i}) \times \sum_{j=1}^{M} ws_L(Sig_{url_j})} \tag{12}$$

We shall recalculate the similarity values using this equation for some of the same examples used earlier. Using WSS_2, session pair ws_K and ws_L are assigned the similarity value of 0.80, while ws_K and ws_R are assigned the value 0.396, implying less similar. In general, most of the sessions contain some identical pages along with a number of different pages. Let us consider an example of such a session pair ws_E={(N_2,0.491,2),(N_7,0.845,1)} and ws_F={(N_3,0.639,2),(N_4,0.599,3),(N_7,0.825,1)}. Both sessions share identical and similar pages, but with low values for structural similarity (see Fig. 4). WSS_1 assigns 0.566 for

these sessions, whereas WSS_2 assigns 0.544, the slightly lower value is the effect of associated significance values.

As another example, consider session pairs ws_G={(N_7,0.439,3),(N_9,0.72,1),(N_{12},0.639,2)} and ws_H={(N_7,0.819,1),(N_{10},0.563,2)}. Both have URLs with high structural similarity (see Fig. 4). WSS_1 assigns 0.342 as opposed to the value of 0.804 assigned by WSS_2. From this we can clearly see that WSS_2 takes into account page pairs with high structural similarity values much better, while WSS_1 does this better for page pairs with lower structural similarity values. Note that in both cases page significance plays a critical role.

Our weighted similarity measure WSS utilizes these properties of both WSS_1 and WSS_2. It uses the maximum score of these two measures to compute a better similarity value among sessions. Equation (13) defines WSS for ws_K and ws_L, where $0<WSS(\text{ws}_K,\text{ws}_L)\leq 1$.

$$WSS(ws_K, ws_L) = \max(WSS_{1KL}, WSS_{2KL}) \tag{13}$$

Our weighted similarity measure $WSS(\text{ws}_K,\text{ws}_L)$ ensures the following properties:

- Nonnegativity: $0<WSS(\text{ws}_K,\text{ws}_L)\leq 1$.
- Identity: $WSS(\text{ws}_K,\text{ws}_K)=1$.
- Symmetry: $WSS(\text{ws}_K,\text{ws}_L)=WSS(\text{ws}_L,\text{ws}_K)$.
- Uniqueness: $WSS(\text{ws}_K,\text{ws}_L)=1$ means $\text{ws}_K=\text{ws}_L$.

In some cases, $WSS(\text{ws}_K,\text{ws}_L)$ may violate Triangle Inequality:

- $WSS(\text{ws}_K,\text{ws}_L)>WSS(\text{ws}_K,\text{ws}_M)+WSS(\text{ws}_M,\text{ws}_L)$.

For Relational Fuzzy Subtractive Clustering (RFSC) algorithm to group sessions, the similarity between ws_K and ws_L is mapped to a distance measure by computing their dissimilarity. Equation (14) defines the the dissimilarity between ws_K and ws_L, where $0\leq WSD(\text{ws}_K,\text{ws}_L)<1$.

$$WSD(ws_K, ws_L) = 1 - WSS(ws_K, ws_L) \tag{14}$$

4.4 Usage Profile Generation

Browsing patterns of web users are highly uncertain and fuzzy in nature. In this respect, fuzzy clustering can be more useful for grouping weighted usage sessions based on their similarity. For clustering sessions, we have chosen the RFSC algorithm because it yields fairly accurate results, is scalable to very large datasets, is reasonably immune to noise present in web data, and is parameter independent [59].

Let, the result of RFSC be denoted by C=$\{C_1,C_2,...,C_Q\}$ the set of Q fuzzy clusters, where each cluster center is an actual weighted session of the dataset, known as cluster prototype. These clusters are needed to be processed further in order to generate usage profiles (or aggregate usage profiles [36]). Basically, usage profiles portray a combined view of subsets of user browsing behaviors based on their interests or preferences, effective for Web personalization [36]. Therefore, for usage profile generation we need to rearrange the pages within each cluster in a manner so that popular pages should come forward and unpopular pages should be pruned out. Each weighted session is a member of all fuzzy clusters with different degree of membership value. Again, each weighted session has a list of accessed pages with their significance weight. Hence, by incorporating the session-wise page significance weight and the cluster-wise membership value of weighted sessions, it is possible to measure popularity of pages in each of the respective clusters. In each usage profile, the most popular page is placed at the top and others are located accordingly. The popularity of url_j in cluster C_Z is computed by equation (15) and is used for generating usage profiles.

$$Popularity[C_Z, url_j] = \frac{\sum_{L=1}^{N} \; ws_L(Sig_{url_j}) \times MV[C_Z, L]}{|N|} \tag{15}$$

Here $0 \leq \text{popularity}(C_Z,\text{url}_j) \leq 1$. $\text{ws}_L(\text{Sig}_{url_j})$ be the significance of url_j in weighted session ws_L and $\text{MV}[C_Z,\text{L}]$ be membership value of ws_L in cluster C_Z.

In the next chapter we describe our adaptation of two recommendation algorithms to include our proposed weighted session similarity measure.

Chapter 5

Recommender Algorithms

Our adaptation of two recommender algorithms based on the model-based collaborative filtering (CF) technique comprises of three changes. Firstly, we use the weighted session similarity measure developed in the previous chapter. Second, we introduce a new parameter "Overlapping ratio", which is used together with the proposed weighted similarity measure for selecting the nearest cluster prototype (s) for the user at recommendation time. Thirdly, we define a new way of defining neighborhood in finding the pages for recommendation. Before we go into details of these changes, we review different collaborative filtering approaches with their properties and limitations.

5.1 A Brief Review of Collaborative Filtering(CF) Algorithms

The goal of a recommender system is either to predict a particular item which is likely to be requested next by an active user (i.e., prediction problem), or to suggest a set of *top-N* items which will be of interest to an active user (i.e., *top-N* recommendation problem). Items can be any type of online information resources including web pages, images, books, and etc. Among all existing recommendation techniques, collaborative filtering(CF) is the most popular and widely used approach. Generally, CF systems rely on gathering and analyzing information about user access patterns and their interests, and suggest a recommendation

list to the users based on preferences of a group of users with similar interests. The recommended items are attached to the last requested page of an active user before sending that page to the user. Basically, the key assumption of CF is that if two users A and B have similar preferences/rating on N-items, then they will have similar preferences/rating for other items [16]. Users may express their preferences or interests on items through explicitly/implicitly rating them. In explicit rating, users may give varying numeric value to items to represent different degree of preferences/interests, whereas implicit user ratings are deduced from user access behavior, for example, visit duration on a web page.

CF systems are either user-centric or item-centric. The user-centric CF systems make use of similarity in user interest while recommending. On the other hand, the item-centric CF systems recommend items to an active user by considering the most similar items, instead of finding similar users. Typically, CF techniques are classified into two major categories [7] based on the underlying search strategy.

- Memory-based Collaborative Filtering(CF) Approach
- Model-based Collaborative Filtering(CF) Approach
- Hybrid Collaborative Filtering(CF) Approach

5.1.1 Memory-Based Collaborative Filtering(CF) Approach

The memory-based CF approach makes use of entire user-item rating database to generate a set of recommendations. It is noted that each active user is assumed to have preferences which are similar to a group of other users. Therefore, by identifying this matching group, it is possible to make predictions or recommendations. This idea is the core of any memory-based CF algorithm. The most common memory-based CF algorithm is the neighborhood-based algorithm [57]. This algorithm includes two major steps, determining similarity between two users or items and producing predictions for the active user by considering weighted average of all ratings on items. Another memory-based algorithm is the top-N recommendation algorithm [57], which can be further user-based or item-based. The general idea in this algorithm is to identify the k most similar users/items using similarity

value, and to aggregate the selected user ratings to choose *top-N* most popular items as recommendation. Therefore, computing the user similarity is a critical step in memory-based CF algorithms. Well-known similarity measures such as Pearson correlation measure and its variations [19, 20, 31, 48, 49, 66], Spearman rank correlation [19, 20] and Cosine-similarity measure [48, 49, 65] are used to compute similarity between users/items.

Many commercial systems such as "www.amazon.com" [27] and "www.CDNow.com" [22], also music and movie recommender systems such as Ringo [53], and Video Recommender [21] utilize memory-based CF approach because of its simple and easy implementation, and its effectiveness for dense datasets. However, this approach has some major shortcomings. Mainly, the memory-based CF approach requires all calculations including the similarity computations and the neighborhood selection to be done in real time, while keeping the entire dataset in memory. As a result, time and memory requirements rise with the number of users and items in a linear fashion. This makes the memory-based CF less scalable for very large datasets. Again, it's performance degrades when data are sparse. To remedy these problems to an extent, model-based CF approach has been evolved.

5.1.2 Model-Based Collaborative Filtering(CF) Approach

A model-based CF technique mainly depends on an access behavioral model developed by applying various data mining or machine learning algorithms on the user-rating dataset. This model allows the systems to learn and to recognize complex patterns from the training dataset, which then assist in making intelligent recommendations. A number of model-based CF algorithms such as Bayesian network models, clustering models, association-rule models, probabilistic latent semantic models, Markov decision processes based models and dependency networks have been explored [7, 18, 22, 35, 58]. For categorical user ratings, classification algorithms are used as CF models, whereas for numeric ratings, regression models and SVD methods show good prediction performance [57].

In a simple Bayesian CF algorithm, a naive Bayes (NB) strategy is used to generate recommendations [33]. In a clustering CF algorithm, a set of clusters, which is a collection of

similar users/items is generated. For recommendation task, these generated clusters need to be analyzed and processed further. In these algorithms, Cosine similarity, Pearson correlation, and etc., can be used to measure similarity between the users/items. The clustering CF algorithms makes the recommendations from a small and highly similar neighborhood (i.e., clusters) rather than the whole dataset, which achieves better scalability. The expensive cluster generation task is done offline. As a result, the online recommendation process can be considerably speeded up. Usually, the regression CF algorithm uses an approximation of the user-ratings to make recommendations based on a regression model [61]. In MDP-based CF algorithms the recommendation task is viewed as a sequential optimization problem, and a Markov decision processes (MDPs) model is used for recommendation purposes [52].

The model-based CF approach can handle sparsity much better than memory-based CF approach. It also has high scalability with large datasets. Further, it improves prediction performance and provides an intuitive rationale for recommendations. However, the shortcomings of model-based CF approach are its expensive model building phase and the potential loss of useful information due to use of a reduced model. Clearly, there are tradeoffs between system scalability and recommendation performance [57].

5.1.3 Hybrid Collaborative Filtering(CF) Approach

Hybrid CF approaches combine CF techniques with other recommendation creation techniques. Typically, content-based filtering systems are incorporated into CF systems in order to achieve better recommendation performance. Popescul et al. [44] proposed a probabilistic model for combining collaborative and content-based recommender systems in order to recommend documents in sparse-data environments. Melville et al. [32] introduced a content-boosted collaborative filtering for making movie recommendations. They used a content-based predictor based on naive Bayes. They replaced the missing values in the sparse user-ratings matrix by the predictions of the content-based predictor, eventually making a pseudo user-rating matrix. Finally, they generated personalized suggestions by

applying the CF approach to this pseudo user-rating matrix. Shahabi et al. [50] developed a recommender system, "Yoda", by merging collaborative filtering and content-based querying to achieve higher accuracy in recommendations.

In some other research work, memory-based and model-based CF algorithms are combined to form a hybrid approach. Generally, this hybrid approach provides better prediction performance than the pure CF approaches. Yu et al. [68] introduced a probabilistic memory-based collaborative filtering (PMCF) by integrating memory-based and model-based CF techniques. They built a mixture model based on a set of stored user profiles and hence used the posterior distribution of the user ratings to make predictions. Similarly, Pennock et al. [43] proposed and evaluated a hybrid CF method called "Personality Diagnosis(PD)" by jointly using memory-based and model-based CF approaches. The PD determined each active user's personality type by computing the probability that he might have the same "personality type" like others and generated the probability with which the user would prefer the new items. In addition, Suryavanshi et al. [60] introduced a fuzzy hybrid CF techniques, which selected the fuzzy nearest prototype(s) for the active user from a set of usage profiles (i.e., behavioral model), and performed the memory-based approach on the selected profile(s) to choose a group of like-minded users to make recommendations. Since the group is small as compared to the entire dataset, it makes the hybrid approach much more scalable than a pure memory-based approach.

5.2 Adaptation of Recommender Algorithms

The performance of a recommender engine, an online component of a personalization system, largely depends on a set of high quality usage profiles and the efficiency of recommender algorithm. In this section, we will discuss our adaptation of recommender algorithms based on the model-based CF approach. We propose adapting of the following:

- Model-based CF with Similarity and Overlapping Ratio
- Fuzzy Hybrid CF with Similarity and Overlapping Ratio

To recommend pages to an active user, it is needed to find the nearest usage profile(s). Mainly, the selected nearest profile signifies similarity in browsing patterns between active user and a group of past users, which is used in making recommendations. In our adaptation, an active usage session us_A is converted into an active weighted session ws_A by estimating page significance and their ranks (using equations (6) to (8)). Let us recall that each cluster is represented by a cluster prototype, which is an actual weighted session of the dataset. Therefore, selection of the nearest profiles requires us to compute similarity between the weighted prototype and ws_A. Our weighted similarity measure is used for doing this computation. An important point to mention is that the page-access duration and access frequency are situation dependent, and play a critical role in computing the page significance. Changes in either of these two parameters alter the page significance value. When using our weighted similarity measure *WSS*, tiny variations in page-significance may lead to a higher similarity for two weighted sessions having more structurally related pages than those with more identical pages. This happens in a few cases. For mitigating this problem, we introduce a new parameter *Overlapping Ratio*.

Definition 1 (Overlapping Ratio)**.** *The overlapping ratio, OR_{A,C_i} between an active weighted session ws_A and a weighted cluster prototype ws_{C_i} is the ratio of the urls they have in common, defined as follows:*

$$OR_{A,C_i} = \frac{|ws_A \cap ws_{C_i}|}{|ws_A|} \tag{16}$$

Both recommender algorithms are modified to combine the similarity (Sim) and the overlapping ratio (OR) for nearest profile selection. From our various experiments, we see that applying this combination to select the nearest profile results in improved recommendation hits in all the recommendation methods used in our experiments (see Chapter 6).

Example 4.

Let ws_{C_1}={(N_5,0.98,1), (N_7,0.78,3), (N_9,0.65,4), (N_{12},0.87,2)} and ws_{C_2}={(N_7,0.64,2), (N_9, 0.89,1), (N_{10},0.45,3)} are two weighted cluster prototypes. Let ws_A={(N_5,0.45,3), (N_7,0.78,

2), (N_9,0.98,1)} be an active weighted session. From our proposed similarity measure (equation (13)), we calculate similarity $WSS(ws_A,ws_{C_1})$=0.822 and $WSS(ws_A,ws_{C_2})$=0.870. We find OR_{A,C_1}=1.0 and OR_{A,C_2}=0.667 (equation (16)). Now, if the nearest cluster selection is based only on similarity value, ws_{C_2} is selected as the nearest prototype for ws_A. But ws_{C_1} and ws_A possess more identical pages, where the page significance values are almost the same. If similarity and overlapping ratio are jointly used to select the nearest cluster prototype, then ws_{C_1} is selected as the nearest one.

5.2.1 Model-Based CF Algorithm with Similarity and Overlapping Ratio

For recommending pages to an active user u_A, the proposed modification to model-based collaborative filtering (CF) algorithm first selects the nearest cluster $C_{nearest}$ using similarity **WSS**(ws_A,ws_{C_z}) between weighted prototype ws_{C_z} and weighted active session ws_A, and their overlapping ratio OR_{A,C_z}. Next, it selects a set of *top_N* most popular urls from $C_{nearest}$ and recommends this list to u_A. The modified algorithm is described below.

Algorithm 1 Model-Based CF with Similarity and Overlapping Ratio

Input: URL=$\{url_1, url_2, ..., url_M\}$ be a set of urls. An active session us_A and a set of clusters C={C_1, C_2,...,C_Q}. Popularity[Q,M] be the cluster-wise url-popularity matrix for all url_i. *NA* be the set of all urls which are not in us_A.
Output: A recommendation list of *top_N* urls.

1: Generate an active weighted session ws_A from us_A.
2: For all clusters $C_Z \in$C, do steps 3, 4 and 5.
3: Calculate WSS(ws_A,ws_{C_z}) between ws_A and ws_{C_z} using equation (13).
4: Calculate OR_{A,C_z} between ws_A and ws_{C_z} using equation (16).
5: Set $Combine_{A,C_z} \leftarrow$ WSS(ws_A,ws_{C_z}) + OR_{A,C_z}.
6: Select nearest cluster $C_{nearest}$ for ws_A with $\max(\forall_{C_z} Combine_{A,C_z})$.
7: For all $url_j \in NA$, recommend *top_N* most popular urls from $C_{nearest}$ by using Popularity[$C_{nearest}$,url_j].

5.2.2 Fuzzy Hybrid CF Algorithm with Similarity and Overlapping Ratio

Our proposed modification to the fuzzy hybrid CF algorithm incorporates basic techniques of both memory-based and model-based CF techniques in order to enhance accuracy and scalability of a recommender engine. In this algorithm, we divide dissimilarity range [0,1]

into R equal sub-ranges (DSR) for each cluster C_Z, and distribute all past weighted sessions ws_L into these sub-ranges using their dissimilarity $\mathbf{WSD}(ws_L, ws_{C_Z})$. Next, we select nearest cluster $C_{nearest}$ for ws_A with the similarity $WSS(ws_A, ws_{C_Z})$ between weighted prototype ws_{C_Z} and weighted active session ws_A, and their overlapping ratio OR_{A,C_z}. After computing dissimilarity $WSD(ws_A, ws_{C_{nearest}})$ between $ws_{C_{nearest}}$ and ws_A from their similarity (equation (14)), we select all sessions which belong to the same DSR as the one to which ws_A belongs. From this set, we select K-most nearest sessions, again using their similarity to ws_A, and compute the popularity of their urls, not yet accessed in ws_A. Finally, a list of *top_N* most popular urls is recommended to the user. The modified algorithm is described below.

Algorithm 2 Fuzzy Hybrid CF with Similarity and Overlapping Ratio

Input: URL=$\{url_1, url_2, ..., url_M\}$ be a set of urls. An active usage session us_A and a set of clusters $C = \{C_1, C_2, ..., C_q\}$. $WSD[Q, N]$ be the dissimilarity matrix between all weighted prototypes ws_{C_z} and all previous weighted sessions ws_L. NA be the set of all urls which are not in us_A. $DSR_{C_z,r}$ be the dissimilarity sub-ranges of C_z with $1 \leq r \leq R$.
Output: A recommendation of *topN* urls.

1: Generate an active weighted session ws_A from us_A.
2: For all clusters $C_z \in$C, do steps 3, 4 and 5.
3: Calculate WSS(ws_A,ws_{C_z}) between ws_A and ws_{C_z} using equation (13).
4: Calculate OR_{A,C_z} between ws_A and ws_{C_z} using equation (16).
5: Set $Combine_{A,C_z} \leftarrow$ WSS(ws_A,ws_{C_z}) + OR_{A,C_z}.
6: Select nearest cluster $C_{nearest}$ for ws_A with $\max(\forall_{C_z} Combine_{A,C_z})$.
7: Set WSD(ws_A,ws_{C_z}) $\leftarrow$ 1 - WSS(ws_A,ws_{C_z}).
8: Select $DSR_{C_{nearest},r}$ by using WSD(ws_A,ws_{C_z}) for ws_A.
9: Choose all previous weighted sessions $ws_{neighbor}$ belong to $DSR_{C_{nearest},r}$ and calculate WSS(ws_A,$ws_{neighbor}$).
10: Select $ws_{K_{nearest}}$ most similar sessions using WSS(ws_A,$ws_{neighbor}$).
11: For all $url_j \in NA$ do step 12
12: For each $ws_{K_{nearest}}$ do step 13
13: If $url_j \in ws_{K_{nearest}}$, then do step 14
14: Set popularity(url_j) $\leftarrow$ WSS(ws_A,$ws_{K_{nearest}}$)$\times$ $ws_{K_{nearest}}(Sig_{url_j})$.
15: Recommend *top_N* most popular urls using popularity(url_j).

In the next chapter we describe the various experiments we have conducted to evaluate the effectiveness of these modifications and compare their performance with a number of other methods popularly used in recommender systems.

Chapter 6

Experimental Results and Performance Evaluation

In this chapter, we describe the series of experiments carried out to evaluate the efficiency and effectiveness of our weighted session similarity (WSS) measure in the context of recommender algorithms, modified as described in the previous chapter. To compare recommendation performance, we also carried out the same experiments with four other similarity measures, namely, Pearson correlation coefficient (PCC), Jaccard coefficient (JC), Cosine similarity (CS) and the measure proposed in [40], which we shall call as Binary Session Similarity (BSS). All experiments were performed on an Intel(R) Xeon(R) 3400 series based workstation running at 2.67 GHz with 4 GB RAM and 454 GB hard disk.

6.1 Dataset and Experimental Setup

For the tests and experiments, we used a user access log from the web server of the Computer Science and Software Engineering department at Concordia University during December 31, 2004 to January 15, 2005. After data cleaning, we had about 46 MB of 200,000 cleaned records. After session extraction, we got 16,816 usage sessions and 12,685 distinct urls. After removing sessions with length of 1 or 2, and computing the page-significance, we had 13,580 weighted sessions with average session length of 7.35 and over 99% of data sparsity,

defined as 1 - $\frac{Nonzero\ Entries}{Total\ Entries}$. We randomly divided the dataset of 13,580 weighted sessions into two parts: the training dataset and the test dataset. We performed the experiments using three different combinations of the training and the test datasets in order to ensure that our recommendation results are not sensitive to a particular partition of dataset. The three different random combinations of the training and the test datasets are as follows:

- Case-1: A training set of 10,864 weighted sessions (80% of total sessions) and a test set of 2716 sessions (20% of total sessions).
- Case-2: A training set of 9506 weighted sessions (70% of total sessions) and a test set of 4074 sessions (30% of total sessions).
- Case-3: A training set of 12,222 weighted sessions (90% of total sessions) and a test set of 1358 sessions (10% of total sessions).

For each test session, some pages are hidden, forming a *Hidden_set*. Our system worked on the training set, and subsequently produced a set of recommendations for the hidden set of each test session. Let *top_N* denote the list of recommended pages. If a hidden page is present in the recommendation list, we call it a *hit*. From the training dataset, we generated the following usage profiles using five different similarity measures and the fuzzy clustering algorithm for each of the selected training sets.

- UP_1=$\{up_{11}, up_{12}, ..., up_{1L}\}$, a set of usage profiles using *BSS* measure.
- UP_2=$\{up_{21}, up_{22}, ..., up_{2R}\}$, a set of usage profiles using *JC* measure.
- UP_3=$\{up_{31}, up_{32}, ..., up_{3X}\}$, a set of usage profiles using *PC* measure.
- UP_4=$\{up_{41}, up_{42}, ..., up_{4Y}\}$, a set of usage profiles using *CS* measure.
- UP_5=$\{up_{51}, up_{52}, ..., up_{5Q}\}$, a set of usage profiles using proposed *WSS* measure.

In our work, the recommender approach utilizing the usage profiles UP_1 and the *BSS* measure for providing recommendations is termed as UP_1 approach. In a similar manner, for other methods listed above we call them as UP_2, UP_3, UP_4, and UP_5 respectively.

Let *NP* denote the number of nearest cluster(s), and *Nearest_K* denote the K-nearest neighbors of each test session. Let *DSR* denote the dissimilarity sub-range. In all experiments, we jointly used the similarity measure and overlapping ratio to select *NP* from the set of usage profiles. However, for selecting *Nearest_K*, we used only the similarity measure. We show the results of experiments by keeping *NP* constant at 1, *Nearest_K* at 100, *DSR* at 0.10, and varying *top_N* to 5, 10, 15, and 20 respectively.

6.2 Performance Evaluation Metrics

We used the metrics, hits, recall, precision and mean reciprocal hit-rank to evaluate effectiveness. For efficiency, we used the recommendation time (in seconds) per user. Each of these metrics is described below:

1. *Hits*: It is the number of items in the *Hidden_set* that are also present in *top_N* recommended items. Therefore, higher the hits, better the system performance in making recommendation.

2. *Recall*: It is defined as the ratio of items in the *Hidden_set* that are correctly recommended. The value of recall is likely to enhance as *top_N* increases. Higher recall value means improved performance. The percentage of *recall*(%) can be defined as follows.

$$Recall(\%) = \frac{|Hidden\_set \cap top\_N|}{|Hiddent\_set|}(\%) \tag{17}$$

3. *Precision*: It shows the ability of recommender systems in producing accurate recommendations. The value of precision is likely to decline as *top_N* increases. Larger value for precision leads to better performance. The percentage of *precision*(%) can be defined as follows.

$$Precision(\%) = \frac{|Hidden\_set \cap top\_N|}{|top\_N|}(\%) \tag{18}$$

Table 7: Total number of clusters for 10,864 training weighted sessions (Case-1).

Usage Profiles	UP_1	UP_2	UP_3	UP_4	UP_5
No. of Clusters	37	33	16	36	68

Table 8: Total number of clusters for 9506 training weighted sessions (Case-2).

Usage Profiles	UP_1	UP_2	UP_3	UP_4	UP_5
No. of Clusters	17	35	17	31	50

Table 9: Total number of clusters for 12,222 training weighted sessions (Case-3).

Usage Profiles	UP_1	UP_2	UP_3	UP_4	UP_5
No. of Clusters	41	37	16	35	78

4. *Mean Reciprocal Hit-Rank (MRHR)*: It assesses the recommendation quality. Earlier occurred hits in *top_N* are given more weight than later occurred hits in *MRHR*. The highest value of *MRHR* is equal to hit-ratio, when all hits are positioned at the first position of *top_N*. However, the lowest value of MRHR is equal to $\frac{hit\_ratio}{|top\_N|}$, when all hits are positioned at the last position of *top_N*. Higher the *MRHR*, better the recommendation quality. Let H be the number of hits which occurred at positions p_1,p_2,.....,p_H in *top_N*. The percentage of *MRHR*(%) can be defined as follows.

$$MRHR(\%) = \frac{1}{|Test\_set|} \sum_{i=1}^{H} \frac{1}{p_i}(\%) \tag{19}$$

6.3 Performance Analysis

We conducted the experiments based on two separate *Hidden_sets*: (1) Most Significant *Hidden_set* and (2) Randomly Selected *Hidden_set*. In first case, the most significant page from each weighted session of the test set was hidden. On the other hand, a randomly selected page from each test session was hidden in the second case. Tables 7, 8, and 9 show the number of clusters obtained from the three different training sets using the five different session similarity measures.

Table 10: Required time (in hours) for the usage profile generation from 10,864 weighted sessions (Case-1).

Usage Profiles	UP_1	UP_2	UP_3	UP_4	UP_5
Required time (*in hours*)	4.0	3.9	2.9	3.8	4.5

Table 11: Two most prominent usage profiles of UP_1, UP_2, and UP_3 for 10,864 weighted sessions (Case-1).

Profile #	UP_1	UP_2	UP_3
1	/ grogono/tunick.html / grogono/tunick-pictures.html / grogono/Photography/ /current_students.html / grogono/photo.html	/ comp238/2005W/ / comp229/ / comp239/2005W/ / comp248/ / comp335/2004F/	/people/people.html /people/faculty.html /current_students.html / chalin/ / chalin/header.html
2	/programs/grad/courses.html /programs/grad/masters/master.html /programs/grad/diploma/courses.html /programs/grad/diploma/comp5511.html /programs/grad/diploma/diploma.html	/ comp445/slides/ / comp445/winter05/ /current_students.html / comp646/winter05/labs/lab1.html / comp646/winter05/labs/	/ comp239/2005W/ / comp248/ / comp229/ / eavis/hobbit/ / eavis/comp249/border.html

Table 12: Two most prominent usage profiles of UP_4 and UP_5 for 10,864 weighted sessions (Case-1).

Profile #	UP_4	UP_5
1	/ comp445/slides/ / comp445/labs/ / comp646/winter05/ / comp646/winter05/labs/ / comp646/winter05/assignments/	/ comp218/ / comp218/Comp218/Comp218WebPage/Slides/ / comp218/Comp218/Comp218WebPage/PDF_Files/ / comp218/Comp218/Comp218WebPage/Html_Files/Main.html / comp218/Comp218/Comp218WebPage/Html_Files/ConU_Logo.html
2	/department/admissions/admissions.html /department/admissions/grad.html /programs/grad/masters/master.html /current_students.html /programs/grad/courses.html	/ soen344/05W/priv/ / soen344/05W/home.html / soen344/05W/references.html / soen344/ / soen344/05W/banner.html

While considering the time for usage profile generation time (i.e., total time for offline tasks), we see that profile generation using the *WSS* measure takes more time. The main reason for this extra time is due to the additional computation involved in obtaining similarity between weighted sessions. In *WSS*, both page significance estimation and the URL-similarity computation are required for measuring similarity. This similarity result is then used as input to the fuzzy clustering algorithm for generating the usage profiles. In comparison, the *BSS* measure requires only estimation of URL-similarity for computing session similarity, assuming equal significance for all accessed pages. The remaining three similarity measures *PCC*, *JC*, and *CS* compute the session similarity based on the identical pages. So, there is no need to compute page similarity, but only the page significance computation. Table 10 shows the usage profile generation time (in hours) using all five similarity measures for the training set of 10,864 weighted sessions (i.e., Case-1). Tables 11

and 12 present the first two most prominent usage profiles from each of UP_1, UP_2, UP_3, UP_4, and UP_5 discovered by RFSC, where only top 5 most popular urls have been shown.

Table 13: Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	715	534	531	723	990
10	881	789	746	887	1206
15	984	925	899	1002	1328
20	1052	1031	978	1067	1406

Table 14: Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	860	797	666	225	1381
10	1104	1122	1019	454	1707
15	1254	1366	1193	600	1893
20	1339	1527	1363	750	2029

Table 15: Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	398	262	199	346	476
10	487	369	327	436	585
15	541	442	417	515	652
20	577	493	475	546	675

6.3.1 Performance Analysis for the Most Significant Hidden_set

In this case, we hide the most significant page, i.e., rank-1 page from each test session. Our recommender algorithms are meant to provide a set of recommendations for each of these hidden pages. We applied five similarity measures *PCC*, *JC*, *CS*, *BSS*, and *WSS* for selecting the nearest clusters and the K-nearest neighbors from their respective usage profiles. We show the results of experiments by keeping *NP* constant at 1, *Nearest_K* at 100, *DSR* at 0.10, and varying *top_N* to 5, 10, 15, and 20 respectively. Comparison of

the recommendation results are shown in terms of *hits*, *recall*, *precision*, *MRHR*, and the recommendation time.

Table 16: Comparison of *MRHR*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	19.22	10.64	10.05	17.25	25.73
10	20.03	11.91	11.01	18.05	26.79
15	20.33	12.30	11.46	18.39	27.14
20	20.47	12.52	11.63	18.53	27.30

Table 17: Recommendaiton time(in seconds) per user for modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with *top_N*=20.

	UP_1	UP_2	UP_3	UP_4	UP_5
top_N	T(sec)	T(sec)	T(sec)	T(sec)	T(sec)
20	0.68	0.65	0.40	0.55	0.75

Tables 13, 14 and 15 present the overall hits obtained using the modified model-based CF method for the three different test sets. For 2716 test sessions (i.e., Case-1), our UP_5 provides increased hits, which are approximately more than 10% to 13% from UP_1, 14% to 17% from UP_2, 16% to 17% from UP_3 and 10% to 13% from UP_4 respectively. Similarly, increased number of hits from UP_5 are also received for other two cases, i.e., Case-2 and Case-3 (with respect to Tables 14 and 15). Figs. 6 and 7 and Table 16 present the comparison of recommendation quality in terms of recall, precision and MRHR for the modified model-based CF method in case of 2716 test sessions (i.e., Case-1). From these figures and table, we find that UP_5 outperforms others by providing recommendations with higher recall, better precision, and superior MRHR at the cost of a negligible increased recommendation time as shown in Table 17. Fig. 6 demonstrates that UP_5 gives 37% of recall at *top_N*=5, which is roughly 10% more than other approaches, and this is further raised to over 50% for *top_N*=20, approximately 12% in excess of others. For the precision metric, similar performance is seen from Fig. 7, where a steady decrease of precision from over 7% at *top_N*=5 to 3% at *top_N*=20 has been observed from UP_5. Again, this is the best

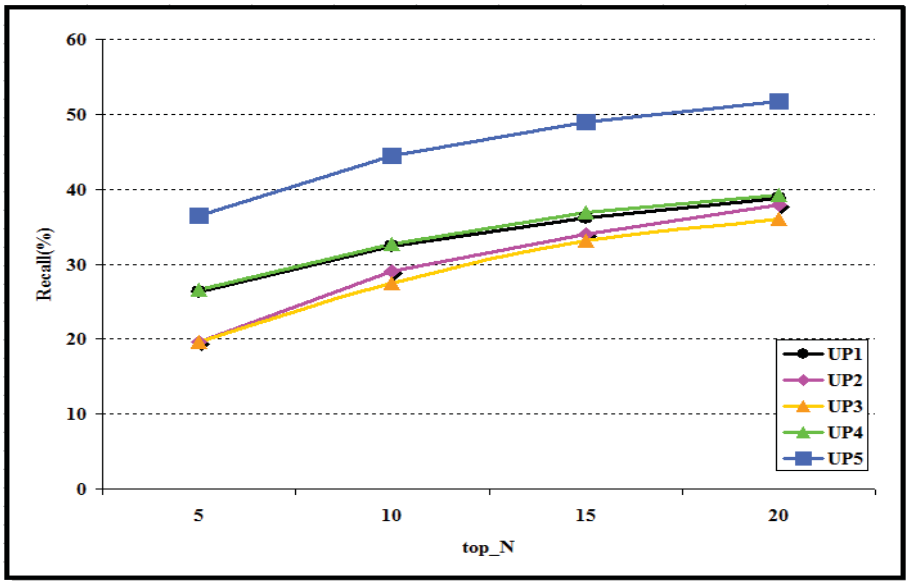


Figure 6: Comparison of *Recall*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

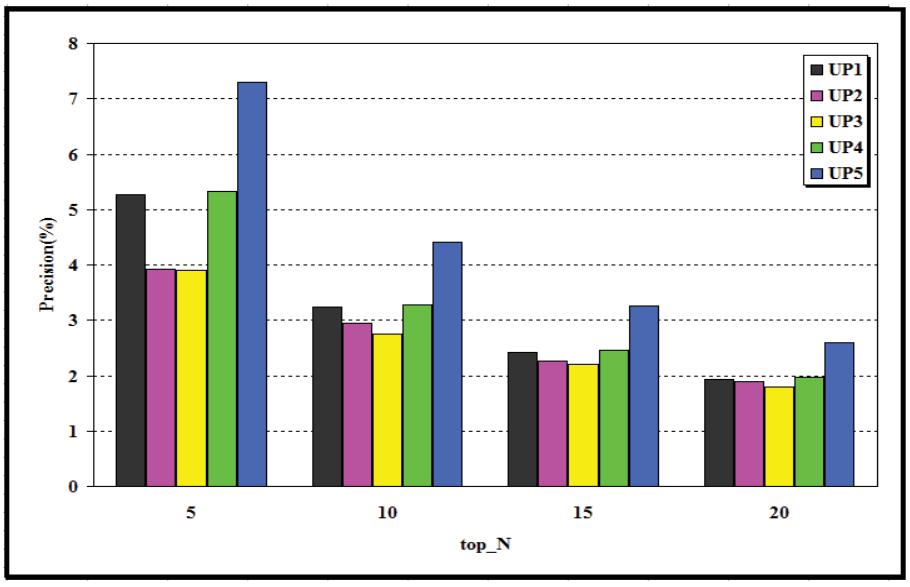


Figure 7: Comparison of *Precision*(%) when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

Table 18: Comparison of results when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	21.1	4.2	14.8	19.6	3.9	9.8	16.4	3.3	8.3	5.5	1.1	2.6	33.9	6.8	23.3
10	27.1	2.7	15.6	27.5	2.8	10.9	25.0	2.5	9.4	11.1	1.1	2.8	41.9	4.2	24.3
15	30.8	2.1	15.9	33.5	2.2	11.4	29.3	2.0	8.0	14.7	1.0	3.1	46.5	3.1	24.7
20	32.9	1.7	16.0	37.5	1.9	11.6	33.5	1.7	10.0	18.4	0.9	3.3	49.8	2.5	24.9

[a]R=Recall,P=Precision,M=MRHR

Table 19: Comparison of results when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	29.3	5.9	21.4	19.3	3.9	10.4	14.7	2.9	6.9	25.5	5.1	17.6	35.1	7.0	23.2
10	35.9	3.6	22.2	27.2	2.7	11.5	24.1	2.4	8.2	32.1	3.2	18.5	43.1	4.3	24.3
15	39.8	2.7	22.6	32.6	2.2	11.9	30.7	2.1	8.7	37.9	2.5	18.9	48.0	3.2	24.7
20	42.5	2.1	22.7	36.3	1.8	12.1	35.0	1.8	8.9	40.2	2.0	19.1	49.7	2.5	24.8

[a]R=Recall,P=Precision,M=MRHR

as compared to others. Table 16 shows the recommendation performance with respect to MRHR for the model-based approach. An important point to note that higher the MRHR, better the recommendation quality. From Table 16, we see that UP_5 gives the maximum rate of MRHR in contrast to others, thereby showing higher recommendation quality. Similarly, better-quality recommendations are obtained from UP_5 as compared to others in each of the other two test sets as well, i.e., 4076 test sessions (i.e., Case-2) and 1358 test sessions (i.e., Case-3) (See Tables 18 and 19).

In case of the modified fuzzy hybrid CF algorithm, the overall recommendation hits obtained for the most significant hidden pages with respect to the three different test sets are shown in Tables 20, 21 and 22 respectively. For 2716 test sessions (i.e., Case-1), UP_5

Table 20: Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	942	854	767	777	1048
10	1122	1049	951	929	1251
15	1211	1158	1043	1015	1394
20	1277	12321	1111	1069	1465

Table 21: Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	1387	1177	1078	409	1560
10	1697	1425	1341	492	1819
15	1883	1598	1471	534	1993
20	1990	1699	1558	569	2110

Table 22: Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	391	375	355	407	524
10	490	458	450	490	640
15	524	514	502	516	700
20	547	554	523	543	729

provides extra hits, which are roughly more than 4% to 7% from UP_1, 7% to 9% from UP_2, 10% to 13% UP_3, and 10% to 15% from UP_4 respectively. In a similar manner, improved recommendation hits from UP_5 are realized as compared to other approaches for Case-2 and Case-3 (with respect to Tables 21 and 22). In addition, Figs. 8 and 9, and Table 23 display the recommendation results in terms of recall, precision and MRHR for the new fuzzy hybrid CF method in case of 2716 test sessions (i.e., Case-1). These tables again show that the overall recommendation performance is improved using this algorithm, independent of the similarity computation method. Fig. 8 shows that the UP_5 yields the best performance for recall with 30% at *top_N*=5, boosted up to 54% at *top_N*=20. Even though the precision of all approaches is increased using this fuzzy hybrid algorithm as compared to the model-based proposal, UP_5 yields results superior to all others (see Fig. 9). In addition, Table 23 shows that higher MRHR is achieved by using UP_5 as compared to others from the fuzzy hybrid CF techniques, thus providing better-quality recommendations. Besides, Table 24 shows that the modified fuzzy hybrid CF algorithm with *BSS* takes the smallest time in generating *top_N*=20 recommendations, while the fuzzy hybrid CF algorithm with *PCC* requires the highest time for generating recommendations. The other approaches also take nearly the same time for *top_N*=20 recommendations. In addition, Tables 25 and 26 show

Table 23: Comparison of *MRHR*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	24.92	22.30	19.26	19.73	27.85
10	25.82	23.35	20.18	20.49	28.86
15	26.08	23.56	20.44	20.74	29.28
20	26.22	23.72	20.59	20.85	29.43

Table 24: Recommendaiton time(in seconds) per user for modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with top_N=20.

top_N	UP_1 T(sec)	UP_2 T(sec)	UP_3 T(sec)	UP_4 T(sec)	UP_5 T(sec)
20	1.61	2.10	4.91	2.13	2.0

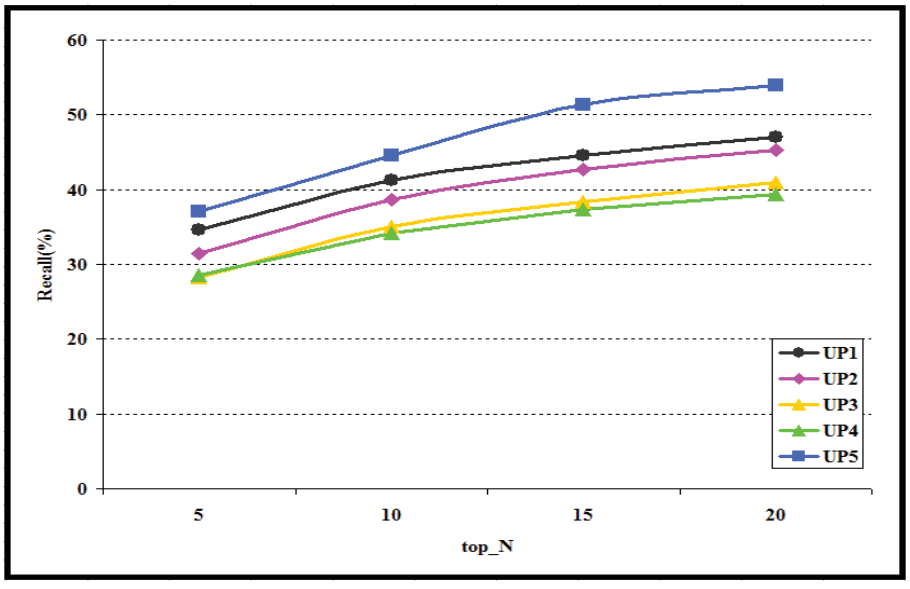


Figure 8: Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

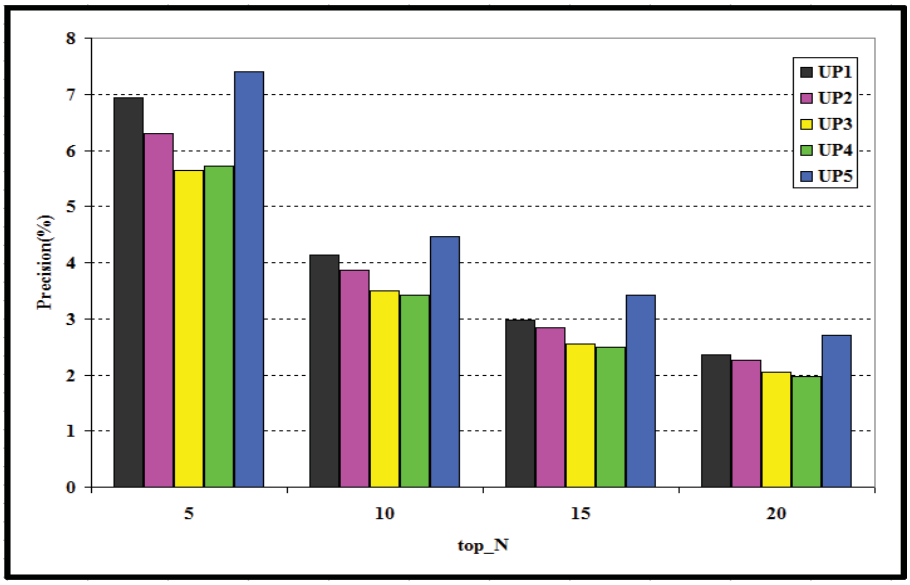


Figure 9: Comparison of *Precision*(%) when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

Table 25: Comparison of results when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 4076 test sessions (Case-2).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	34.1	6.8	24.3	28.9	5.8	20.0	26.5	5.3	17.7	14.0	2.8	7.4	37.6	7.5	26.2
10	41.7	4.2	25.3	35.0	3.5	20.8	32.9	3.3	18.6	13.1	1.3	7.3	45.9	4.6	27.3
15	46.2	3.1	25.7	39.2	2.6	21.2	36.1	2.4	18.8	12.1	0.8	7.2	49.9	3.3	27.7
20	48.9	2.5	25.8	41.7	2.1	21.3	38.2	1.9	18.9	10.1	0.5	6.7	52.5	2.6	27.8

[a]R=Recall,P=Precision,M=MRHR

Table 26: Comparison of results when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 1358 test sessions (Case-3).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	30.3	6.1	22.2	27.6	5.5	18.6	26.1	5.2	15.9	30.0	6.0	21.4	38.6	7.7	26.7
10	37.1	3.7	23.2	33.7	3.4	19.4	33.1	3.3	16.8	36.1	3.6	22.3	47.1	4.7	27.8
15	40.0	2.0	23.4	37.9	2.5	19.7	37.0	2.5	17.1	38.0	2.5	22.4	51.6	3.4	28.2
20	41.3	2.1	23.5	40.8	2.0	19.9	38.5	1.9	17.2	40.0	2.0	22.5	53.7	2.7	28.3

[a]R=Recall,P=Precision,M=MRHR

the recommendation performance in case of the modified fuzzy hybrid CF algorithm for the other two test sets, i.e., 4076 test sessions (i.e., Case-2) and 1358 test sessions (i.e., Case-3). It is important to mention here that UP_4 presents the second best performance in terms of hit, recall, precision, and MRHR for recommending the most-significant hidden pages in model-based CF technique, whereas UP_1 takes the position after UP_5 with fuzzy hybrid CF technique.

6.3.2 Performance Analysis for the Randomly Selected Hidden_set

In this case, we hide a randomly selected page from each test session. Our recommender algorithms provide *top_N* recommendations for this hidden set. We show the results of experiments by keeping *NP* constant at 1, *Nearest_K* at 100, *DSR* at 0.10, and varying *top_N* to 5, 10, 15, and 20 respectively. Tables 27, 28, and 29 show the overall performance for the random *Hidden_set* of 2716 test sessions in terms of *hits*, *recall*, *precision*, and *MRHR* for the modified model-based CF and fuzzy hybrid CF approaches respectively. From these tables, we observe that our UP_5 provides a better recommendation quality with respect to all others for randomly selected *Hidden_set*.

Each hidden page posseses a significance weight in the range $[0, 1]$. We divide this range into the following three sub-ranges:

- "High significance" range from 0.41 to 1.0
- "Mid significance" range from 0.11 to 0.40
- "Low significance" range from 0.0 to 0.10

Further, we rank the pages of the test sessions based on their significance. The most significant page possesses the rank 1 and the remaining pages are ranked accordingly. It has been mentioned that the average weighted session length is 7.35. It is reasonable to consider that the pages belonging to half of the average length of a session as high-ranked pages, while the rests are low-ranked. We consider the pages with rank 1, 2, and 3 as "High-rank" pages. Our target is to find out how well the *high-significant* and also the *high-ranked*

Table 27: Comparison of overall *Hits* when using modified model-based and fuzzy CF algorithms for randomly selected *Hidden_set* from 2716 test sessions (Case-1).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	771	435	273	667	873	937	680	669	862	1020
10	960	578	352	819	1038	1124	863	864	1029	1245
15	1042	680	439	936	1155	1181	957	939	1112	1363
20	1128	770	509	1001	1236	1261	1039	993	1184	1448

Table 28: Comparison of results when using modified model-based CF algorithm for randomly selected *Hidden_set* from 2716 test sessions (Case-1).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	28.4	5.7	22.1	16.0	3.2	8.7	10.1	2.0	5.2	24.6	4.9	15.1	32.1	6.4	22.3
10	35.4	3.5	23.0	21.3	2.1	9.4	13.0	1.3	5.6	30.2	3.0	15.8	38.2	3.8	23.3
15	38.4	2.6	23.3	25.1	1.7	9.7	16.2	1.1	5.8	34.5	2.3	16.1	42.5	2.8	23.5
20	41.5	2.1	23.5	28.4	1.4	9.9	18.7	0.9	6.0	36.9	1.9	16.3	45.5	2.3	23.7

[a]R=Recall,P=Precision,M=MRHR

Table 29: Comparison of results when using modified fuzzy hybrid CF algorithm for randomly selected *Hidden_set* from 2716 test sessions (Case-1).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	34.5	6.9	25.8	25.0	5.0	17.7	24.6	4.9	15.8	31.7	6.3	21.5	37.6	7.5	26.4
10	41.4	4.1	26.9	31.8	3.2	18.6	31.8	3.2	16.7	37.9	3.8	22.3	45.8	4.6	27.3
15	43.5	2.9	27.2	35.2	2.4	18.9	34.6	2.3	17.0	40.9	2.7	22.6	50.2	3.4	27.5
20	46.4	2.3	27.4	38.3	1.9	19.0	36.6	1.8	17.1	43.6	2.2	22.7	53.3	2.7	27.6

[a]R=Recall,P=Precision,M=MRHR

pages are recommended, as ideally, they should not be missed by any recommender system. We explain the experimental results with respect to the following two perspectives.

1. Recommending *high-significant* hidden pages

2. Recommending *high-ranked* hidden pages

6.3.2.1 Performance Analysis of Recommending the *High-Significant* Pages

Out of randomly selected 2716 hidden pages (i.e., Case-1), a total of 1122 pages is identified as *high-significant* pages. On the other hand, 1638 pages are found as *high-significant* for Case-2, where the total hidden pages are 4074. In addition, we discover 554 *high-significant* pages from a total of 1358 hidden pages in Case-3. Tables 30, 31, and 32 demonstrate the

Table 30: Comparison of *Hits* for 1122 randomly selected *high-significant* hidden pages (Case-1).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	302	234	141	332	426	410	354	330	358	485
10	368	310	182	385	506	485	436	405	412	564
15	398	366	227	428	558	523	468	441	439	607
20	436	408	271	457	579	546	496	464	465	639

Table 31: Comparison of *Hits* for 1638 randomly selected *high-significant* hidden pages (Case-2).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	358	317	120	79	558	564	500	411	158	657
10	444	453	166	167	684	688	577	510	198	805
15	487	524	184	219	752	756	643	569	209	884
20	524	580	222	274	802	802	693	605	218	943

Table 32: Comparison of *Hits* for 554 randomly selected *high-significant* hidden pages (Case-3).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	172	91	39	146	206	191	150	131	183	244
10	212	123	67	176	246	228	176	166	217	294
15	229	153	82	206	268	248	198	181	225	309
20	242	168	99	223	279	261	214	196	231	325

overall hits for the *high-significant* hidden pages using the modified model-based and the modified fuzzy hybrid CF approaches for the three respective cases. From these tables, it can be seen that on an average the fuzzy hybrid method provides better hit rate as compared to the model-based CF in all cases. For the 1122 hidden *high-significant* pages (i.e., Case-1), Table 30 shows that in the modified model-based CF algorithm UP_5 gives extra hits of nearly 10% to 14% from UP_1, 15% to 17% from UP_2, 20% to 29% from UP_3, and 8% to 11% from UP_4 respectively. UP_4 offers the second highest hit-rates after UP_5. On an average, more than 7% hits from UP_1, 12% from UP_2, and 14% from both of UP_3 and UP_4 are achieved by the modified fuzzy hybrid CF with UP_5. UP_1 takes the position

immediately after UP_5 with respect to hit-rate in the fuzzy hybrid approach. In a similar manner, we can see that UP_5 provides the best hit-rate as compared to others using both of the CF algorithms for Case-2 and Case-3 (see Tables 31 and 32).

Table 33: Comparison of *MRHR*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	21.19	10.76	6.44	19.03	26.25
10	22.0	11.65	6.91	19.66	27.25
15	22.19	12.05	7.21	19.97	27.62
20	22.39	12.27	7.43	20.12	27.73

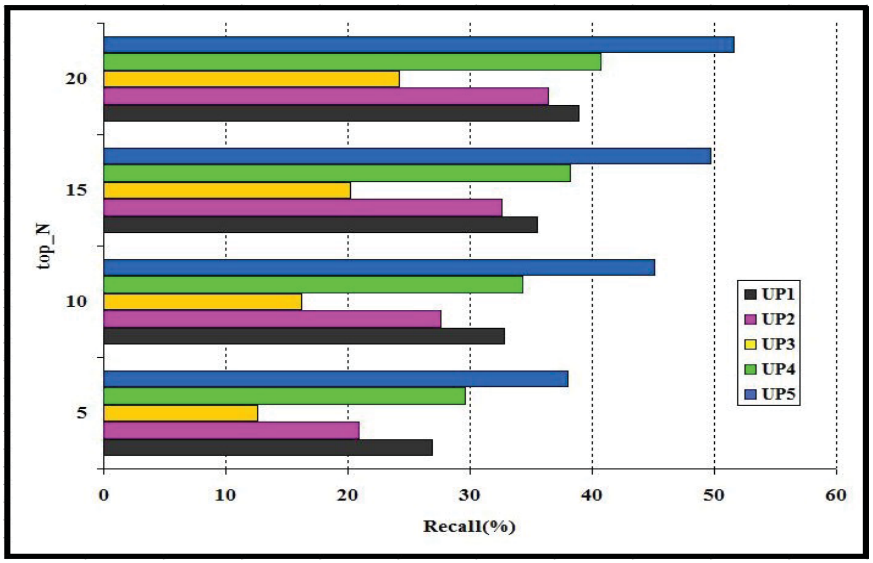


Figure 10: Comparison of *Recall*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

Figs. 10 and 11 show the quality of recommendations in terms of recall and precision obtained using the modified model-based CF system for a total of 1122 random *high-significant* pages (i.e., Case-1). Fig. 10 shows that UP_5 yields better recall as compared to the rest by giving roughly 38% at *top_N*=5, and raised to almost 52% at *top_N*=20. Again in case of precision (see Fig. 11), UP_5 obtains over 7% at *top_N*=5 and 2.58% at *top_N*=20, which are the highest precisions in terms of their respestive recommendation size among

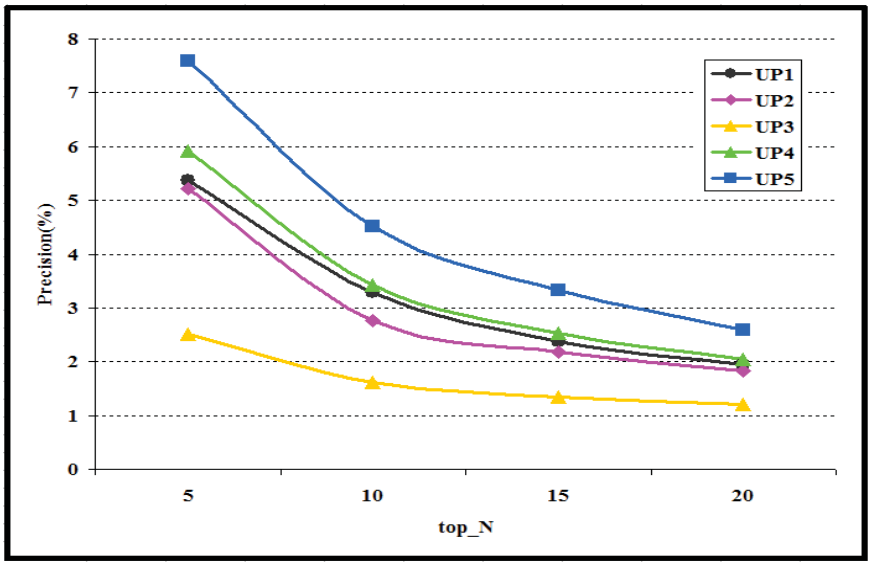


Figure 11: Comparison of *Precision*(%) when using modified model-based CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

all methods. Further, Table 33 confirms that UP_5 provides better-quality recommenations for the modified model-based method by giving larger MRHR. By observing the overall performance of the modified model-based CF method for the 1122 random *high-significant* pages (i.e., Case-1), it is found that after UP_5, the second best performance is obtained from UP_4 in terms of hit, recall and precision, and UP_1 in terms of MRHR (i.e., Case-1). From Tables 34 and 35, we can also observe the enhanced recommendation performance of UP_5 as compared to the rest in other two cases (i.e., Case-2 and Case-3).

Table 34: Comparison of results when using modified model-based CF algorithm for 1638 randomly selected *high-significant* hidden pages (Case-2).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	21.9	4.4	15.7	19.4	3.9	10.0	7.3	1.5	3.0	4.8	1.0	1.7	44.1	8.8	30.8
10	27.1	2.7	16.4	27.7	2.8	11.1	10.1	1.0	3.4	10.2	1.0	2.4	53.1	5.3	32.0
15	29.7	2.0	16.6	32.0	2.1	11.4	11.2	0.8	3.4	13.4	0.9	2.7	55.8	3.7	32.2
20	32.0	1.6	16.7	35.4	1.8	11.6	13.6	0.7	3.6	17.4	0.9	3.6	58.7	2.9	32.4

[a]R=Recall,P=Precision,M=MRHR

Table 35: Comparison of results when using modified model-based CF algorithm for 554 randomly selected *high-significant* hidden pages (Case-3).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	31.1	6.2	23.4	16.4	3.3	9.4	7.0	1.4	5.1	26.4	5.3	17.2	37.2	7.4	25.0
10	38.3	3.8	24.4	22.2	2.2	10.3	12.1	1.2	5.6	31.8	3.2	18.0	44.4	4.4	25.9
15	41.3	2.8	24.6	27.6	1.8	10.7	14.8	1.0	5.9	37.2	2.5	18.4	48.4	3.2	26.2
20	43.7	2.2	24.7	30.3	1.5	10.8	17.9	0.9	6.1	40.3	2.0	18.6	50.4	2.5	26.3

[a]R=Recall,P=Precision,M=MRHR

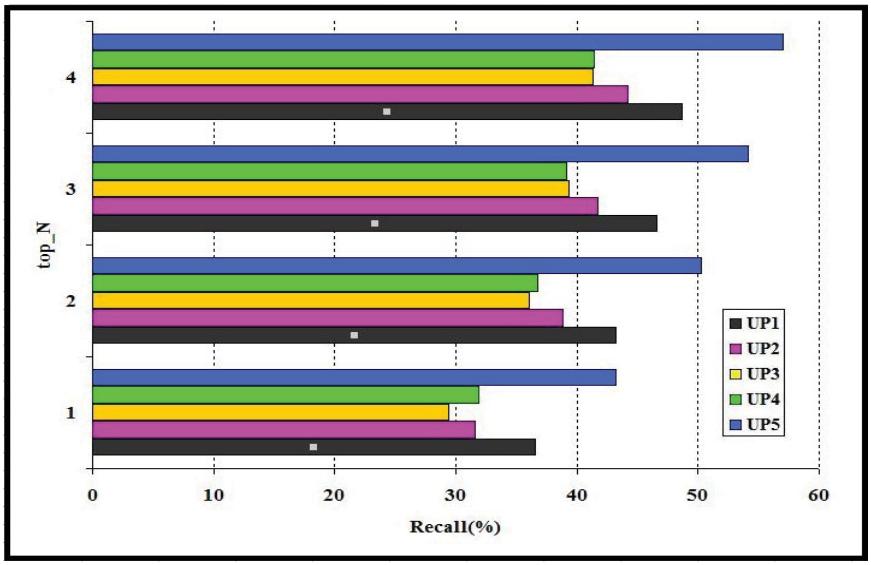


Figure 12: Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

Figs. 12 and 13, and Table 36 present the recommendation performance in terms of recall, precision, and MRHR respectively for the modified fuzzy hybrid CF algorithm for 1122 randomly hidden *high-significant* pages (i.e., Case-1). Fig. 12 shows that the obtained recall is over 43% at *top_N*=5, increased to 57% at *top_N*=20 in case of the modified fuzzy hybrid CF method with UP_5, making it superior to all the other methods. In addition, a precision of nearly 9% at *top_N*=5, smoothly decreased to nearly 3% at *top_N*=20 are obsvered in case of UP_5 from Table 13. An important point to note is that the precison results received using UP_5 for different settings of *top_N* are the highest as compared to others, showing better quality recommendations. Furthermore, Table 36 confirms that higher recommendation quality with the highest MRHR is achieved from modified fuzzy hybrid

Table 36: Comparison of $MRHR(\%)$ when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

top_N	UP_1	UP_2	UP_3	UP_4	UP_5
5	28.21	24.04	20.55	23:80	32.33
10	29.11	24.99	21.44	24:42	33.27
15	29.37	25.22	21.69	24:63	33.58
20	29.49	25.36	21.80	24:77	33.74

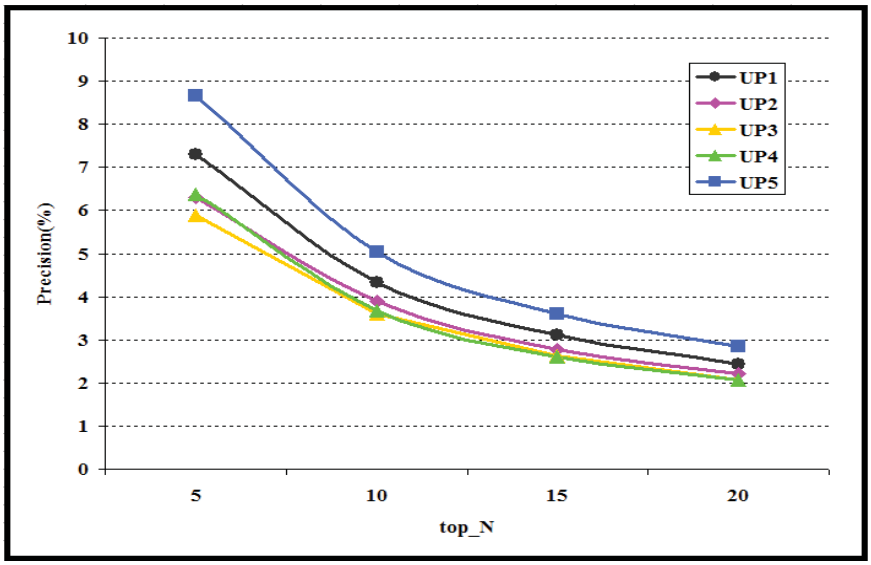


Figure 13: Comparison of *Precision*(%) when using modified fuzzy hybrid CF algorithm for 1122 randomly selected *high-significant* hidden pages (Case-1).

CF using our UP_5. UP_1 gives the second best performance with the modified fuzzy hybrid CF technique as compared to UP_2, UP_3, and UP_4. In a similar manner, Tables 37 and 38 present the recommendations results of UP_5 with improved quality in case of the modified fuzzy hybrid CF approach for Case-2 and Case-3 respectively.

Table 37: Comparison of results when using modified fuzzy hybrid CF algorithm for 1638 randomly selected *high-significant* hidden pages (Case-2).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	34.4	6.9	24.8	30.5	6.1	22.4	25.1	5.0	17.6	9.7	1.9	6.8	40.1	8.0	28.4
10	42.0	4.2	25.8	35.2	3.5	23.1	31.1	3.1	18.5	12.3	1.2	7.1	49.2	4.9	29.6
15	46.2	3.1	26.1	39.3	2.6	23.4	34.7	2.3	18.7	12.7	0.9	7.2	54.0	3.6	29.9
20	49.0	2.5	26.3	42.3	2.1	23.5	36.9	1.9	18.9	13.1	0.7	7.2	57.6	2.9	30.1

[a]R=Recall,P=Precision,M=MRHR

Table 38: Comparison of results when using modified fuzzy hybrid CF algorithm for 554 randomly selected *high-significant* hidden pages (Case-3).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	34.5	6.9	26.7	27.1	5.4	19.1	23.7	4.7	14.7	33.0	6.6	23.8	44.1	8.8	30.8
10	41.2	4.1	27.6	31.8	3.2	19.7	30.0	3.0	15.5	39.2	3.9	24.6	53.1	5.3	32.0
15	44.8	3.0	27.9	35.7	2.4	20.0	32.7	2.2	15.7	40.6	2.7	24.8	55.8	3.7	32.2
20	47.1	2.4	28.0	38.6	1.9	20.1	35.4	1.8	15.9	41.7	2.1	24.8	58.7	2.9	32.4

[a]R=Recall,P=Precision,M=MRHR

6.3.2.2 Performance Analysis of Recommending the *High-Ranked* Pages

In Case-1, a total of 1661 pages is identified as *high-ranked* pages from randomly selected 2716 hidden pages. On the other hand, 2476 pages and 811 pages are discovered as *high-ranked* for Case-2 (i.e., from a total of 4074 hidden pages) and Case-3 (i.e., from a total of 1358 hidden pages) respectively.

Table 39: Comparison of *Recall*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	28.24	20.05	12.28	28.72	36.61	37.93	30.70	28.66	34.80	43.41
10	35.28	26.61	15.95	33.71	43.41	45.39	38.47	35.46	40.28	51.48
15	38.05	30.58	19.50	38.41	48.10	48.46	41.72	38.89	42.87	56.0
20	41.48	34.92	22.52	41.06	51.05	50.87	44.61	41.0	45.52	59.0

Table 40: Comparison of *Precision*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	5.65	4.01	2.46	5.74	7.30	7.59	6.14	5.73	6.96	8.68
10	3.53	2.66	1.60	3.37	4.34	4.54	3.85	3.55	4.03	5.15
15	2.54	2.04	1.30	2.56	3.21	3.23	2.78	2.59	2.86	3.73
20	2.07	1.75	1.12	2.05	2.55	2.54	2.23	2.05	2.28	2.95

Table 41: Comparison of *MRHR*(%) when using modified model-based and fuzzy hybrid CF algorithms for 1661 randomly selected *high-ranked* hidden pages (Case-1).

	Model-based CF Algorithm					Fuzzy Hybrid CF Algorithm				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	21.74	10.91	6.30	17.33	25.27	29.10	22.53	18.96	24.44	31.10
10	22.68	11.79	6.75	17.97	26.21	30.11	23.53	19.89	25.18	32.18
15	22.90	12.11	7.02	18.34	26.59	30.36	23.79	20.15	25.39	32.54
20	23.09	12.35	7.19	18.50	26.76	30.50	23.95	20.27	25.54	32.71

Tables 39, 40, and 41 mutually show the quality of recommendations obtaining from the new model-based and the new fuzzy hybrid CF algorithms with all similarity approaches for recommending *high-ranked* hidden pages in terms of recall, precision, and MRHR respectively. From these tables, it can be observed that both of model-based and fuzzy hybrid CF method with UP_5 provide the best recommendation quality as compared to all others by giving higher recall, superior precision, and larger MRHR. Looking at the rest, the next best performances are seen from UP_2 and UP_4 from both of the modified recommender algorithms. In addition, Tables 42 and 43 present the recommendation performance of the modified model-based CF algorithms for the hidden *high-rank* pages of Case-2 and Case-3 respectively. From these tables, we can see that UP_5 shows better-quality performance in recommending *high-rank* hidden pages as compared to others. The same improved trend can be observed using the modified fuzzy hybrid CF algorithm with UP_5 from Tables 44 (i.e., for Case-2) and 45 (i.e., for Case-3) respectively.

Table 42: Comparison of results when using modified model-based CF algorithm for 2476 randomly selected *high-rank* hidden pages (Case-2).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	22.9	4.6	16.6	18.0	3.6	9.6	8.4	1.7	3.2	6.9	1.4	2.5	42.2	8.4	28.1
10	29.2	2.9	17.5	25.5	2.6	10.6	11.0	1.1	3.5	11.8	1.2	3.2	53.3	5.3	29.5
15	31.9	2.1	17.7	29.9	2.0	10.9	12.0	0.8	3.6	14.5	1.0	3.4	57.6	3.8	29.9
20	34.4	1.7	17.8	33.2	1.7	11.1	14.3	0.7	3.7	16.7	0.8	2.9	60.4	3.0	30.0

[a]R=Recall,P=Precision,M=MRHR

Table 43: Comparison of results when using modified model-based CF algorithm for 811 randomly selected *high-rank* hidden pages (Case-3).

Model-based CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	31.0	6.2	22.7	16.8	3.4	9.5	8.6	1.7	6.6	24.9	5.0	15.2	35.5	7.1	23.1
10	39.2	3.9	23.9	22.6	2.3	10.2	12.3	1.2	7.1	31.8	3.2	16.2	41.9	4.2	24.0
15	43.0	2.9	24.2	27.5	1.8	10.7	14.8	1.0	7.2	37.1	2.5	16.6	47.1	3.1	24.4
20	45.8	2.3	24.3	30.2	1.5	10.8	17.0	0.9	7.4	39.6	2.0	16.7	50.0	2.5	24.5

[a]R=Recall,P=Precision,M=MRHR

Table 44: Comparison of results when using modified fuzzy hybrid CF algorithm for 2476 randomly selected *high-rank* hidden pages (Case-2).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	36.2	7.2	26.0	27.5	5.5	19.1	23.6	4.7	15.5	9.6	1.9	6.7	40.9	8.2	27.9
10	43.4	4.3	27.0	34.1	3.4	19.9	30.3	3.0	16.5	12.1	1.2	6.9	50.4	5.0	29.2
15	47.8	3.2	27.3	38.5	2.6	20.3	33.9	2.3	16.7	12.7	0.9	6.9	55.0	3.7	29.5
20	50.2	2.5	27.4	41.7	2.1	20.5	36.1	1.8	16.9	13.3	0.7	7.0	58.4	2.9	29.7

[a]R=Recall,P=Precision,M=MRHR

Table 45: Comparison of results when using modified fuzzy hybrid CF algorithm for 811 randomly selected *high-rank* hidden pages (Case-3).

Fuzzy Hybrid CF Algorithm[a]															
	UP_1			UP_2			UP_3			UP_4			UP_5		
top_N	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%	R%	P%	M%
5	32.8	6.6	24.8	27.3	5.5	17.7	22.0	4.4	13.7	33.1	6.6	22.7	42.2	8.4	28.1
10	41.4	4.1	26.1	33.4	3.3	18.5	28.9	2.9	14.7	40.6	4.1	23.8	53.3	5.3	29.5
15	45.3	3.0	26.4	37.1	2.5	18.8	31.8	2.1	14.9	42.8	2.9	24.0	57.6	3.8	29.9
20	47.2	2.4	26.5	39.7	2.0	19.0	34.4	1.7	15.0	44.3	2.2	24.1	60.4	3.0	30.0

[a]R=Recall,P=Precision,M=MRHR

6.3.3 Impact of Number of Nearest Clusters

For recommending to an active weighted session, it is possible to select more than one cluster prototype as nearest (i.e., *NP*). Tables 46 shows the total hits obtained from the modified model-based CF algorithm for the most-significant hidden pages in Case-1 (i.e., a total of 2716 test sessions) with all five similarity computation methods. In addition, Table 47 presents the total recommendation hits for the random hidden pages in Case-2 (i.e., a total of 4074 test sessions) from the modified model-based CF algorithm. From these tables, it is observed that UP_1 gives recommendation hits, which decreases with increasing values of *NP*. In contrast, UP_5 provides enhanced hits with increasing *NP*. The recommendation hits obtained from UP_2, UP_3, and UP_4 may vary with *NP*. It is important to mention that in these experiments we have considered only one prototype as nearest for the modified fuzzy hybrid CF algorithm, as the time required for generating the recommendation becomes more with increasing number of *NP*.

Table 46: Comparison of *Hits* when using modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with *NP*=1 and 2.

U.P.	UP_1		UP_2		UP_3		UP_4		UP_5	
N.P.	*NP*		*NP*		*NP*		*NP*		*NP*	
top_N	1	2	1	2	1	2	1	2	1	2
5	715	590	534	521	531	504	723	747	990	1028
10	881	809	789	742	746	712	887	940	1206	1247
15	984	970	925	911	899	882	1002	1038	1328	1389
20	1052	1048	1031	1030	978	971	1067	1127	1406	1473

Table 47: Comparison of *Hits* when using modified model-based CF algorithm for the randomly selected *Hidden_set* from 4076 test sessions (Case-2) with *NP*=1 and 2.

U.P.[a]	UP_1		UP_2		UP_3		UP_4		UP_5	
N.P.[b]	*NP*		*NP*		*NP*		*NP*		*NP*	
top_N	1	2	1	2	1	2	1	2	1	2
5	956	889	1144	1151	307	379	262	260	1144	1152
10	1197	1141	1399	1414	387	435	408	400	1399	1414
15	1308	1299	1549	1564	428	562	496	492	1549	1564
20	1394	1358	1670	1704	525	663	581	562	1670	1704

[a]Usage profiles
[b]Nearest prototypes

Next we discuss the impact of the number of nearest neighbors in the modified fuzzy hybrid CF Algorithm. After selecting the nearest prototype, it is required to choose the nearest neighbors for the active sessions i.e., *Nearest_K*. Table 48 shows the recommendation hits achieved from the modified fuzzy hybrid CF algorithm for the most-significant hidden pages in Case-1 (i.e., a total of 2716 test sessions), whereas Tables 49 presents the results for the random hidden pages in Case-3 (i.e., a total of 1358 test sessions). From these tables, it is observed that increasing number of *Nearest_K* results in a drop in the overall hits with UP_1 and UP_5. In contrast, enhanced recommendation hits are perceived for UP_2 and UP_3 with increasing *Nearest_K*. UP_4 provides almost similar results with varying *Nearest_K* in case all cases. As also mentioned earlier time required for generating the recommendation increases highly with increased number of *Nearest_K*.

Table 48: Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1) with *Nearest_K*=100 and 200.

U.P.	UP_1		UP_2		UP_3		UP_4		UP_5	
N.N.	*Nearest_K*		*Nearest_K*		*Nearest_K*		*Nearest_K*		*Nearest_K*	
top_N	100	200	100	200	100	200	100	200	100	200
5	942	873	854	860	767	851	777	778	1048	1006
10	1122	1059	1049	1059	951	1011	929	933	1251	1212
15	1211	1143	1158	1164	1043	1110	1015	1021	1394	1361
20	1277	1198	1232	1239	1111	1178	1069	1072	1465	1430

Table 49: Comparison of *Hits* when using modified fuzzy hybrid CF algorithm for the randomly selected *Hidden_set* from 1358 test sessions (Case-3) with Nearest_K=100 and 200.

U.P.[a]	UP_1		UP_2		UP_3		UP_4		UP_5	
N.N.[b]	*Nearest_K*		*Nearest_K*		*Nearest_K*		*Nearest_K*		*Nearest_K*	
top_N	100	200	100	200	100	200	100	200	100	200
5	449	431	297	305	277	294	435	433	525	494
10	562	541	379	386	362	385	538	534	654	633
15	607	577	442	451	402	432	580	578	718	700
20	634	606	480	490	441	471	606	605	755	738

[a]Usage profiles
[b]K-nearest neighbors

6.3.4 Impact of DSR in the Modified Fuzzy Hybrid Collaborative Filtering Algorithm

The dissimilarity sub-ranges (i.e., *DSR*) used in the modified fuzzy hybrid CF algorithm plays a critical role in *Nearest_K* selection and also in recommendation performance. A very low value of *DSR* implies a very small group of neighbor sessions for the active sessions, which decreases the recommendation time but with a corresponding reduction recommendation hits. A higher value of *DSR* gives larger number of neighbors, which in turn gives better hit rates, but at the cost of increase in recommendation time. This is the case for all similarity methods we used in the experiments including the UP_5. Tables 50 and 51 present the effect of *DSR* while recommending the *high-significant* pages in terms of recall and recommendation time, where the total test sessions are 2716 (i.e., Case-1).

Table 50: Comparison of *Recall*(%) when using modified fuzzy hybrid CF algorithm for *high-significant* pages from 2716 test sessions (Case-1) with DSR=0.10 and 0.05.

Fuzzy Hybrid CF Algorithm										
DSR^a	0.10					0.05				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	36.54	31.55	29.41	31.91	43.23	32.81	29.55	28.52	30.21	40.65
10	43.23	38.86	36.10	36.72	50.27	38.23	35.57	34.23	34.76	47.52
15	46.61	41.71	39.31	39.13	54.10	40.76	39.0	38.51	36.81	51.45
20	48.66	44.21	41.36	41.44	57.0	43.0	41.12	40.91	40.91	53.67

[a]Dissimilarity sub-range

Table 51: Comparison of recommendation time(in seconds) per user when using modified fuzzy hybrid CF algorithm for *high-significant* pages from 2716 test sessions (Case-1) with *DSR*=0.10 and 0.05.

Fuzzy Hybrid CF Algorithm										
DSR^a	0.10					0.05				
	Time(in second)					Time(in second)				
top_N	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
20	1.61	2.10	4.91	2.13	2.0	1.05	1.37	1.43	3.24	1.29

[a]Dissimilarity sub-range

6.3.5 Impact of Overlapping Ratio in Modified Model-based and Fuzzy Hybrid Collaborative Filtering Algorithms

It was already mentioned that overlapping ratio (i.e., OR) is utilized with the similarity measure in order to select the nearest cluster prototype. This combination is mainly used to avoid certain situations, where the nearest cluster selection is biased with more structurally related pages, instead of more identical pages but with small variation in page significance. Fig. 52 shows the impact of using the similarity measure and the overlapping ratio jointly in the modified model-based CF algorithm for the most significant hidden pages in case of 2716 test sessions (i.e., Case-1). In addition, Fig. 53 shows the impact of joint use of the similarity measure and the overlapping ratio in the modified fuzzy hybrid CF algorithm for the randomly selected hidden pages in case of 4074 test sessions (i.e., Case-2). Both of these tables confirm that enhanced recommendation hits are obtained using this combination, instead of using only the similarity measure for the nearest prototype selection. Moreover, the required recommendation times are not increased significantly by the consideration of this combination.

Table 52: Impact of similarity measure and overlapping ratio in modified model-based CF algorithm for the Most Significant *Hidden_set* from 2716 test sessions (Case-1).

Model-based CF Algorithm										
top_N	similarity measure only					similarity measure and overlapping ratio jointly				
	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	686	514	507	701	947	715	534	531	723	990
10	850	768	702	855	1167	881	789	746	887	1206
15	945	909	839	972	1290	984	925	899	1002	1328
20	1013	1009	918	1041	1376	1052	1031	978	1067	1406

Table 53: Impact of similarity measure and overlapping ratio in modified fuzzy hybrid CF algorithm for randomly selected *Hidden_set* from 4074 test sessions (Case-2).

Fuzzy Hybrid CF Algorithm										
top_N	similarity measure only					similarity measure and overlapping ratio jointly				
	UP_1	UP_2	UP_3	UP_4	UP_5	UP_1	UP_2	UP_3	UP_4	UP_5
5	1387	877	765	340	1437	1401	909	818	359	1468
10	1666	1134	1003	437	1808	1678	1171	1079	458	1830
15	1832	1307	1125	476	1984	1840	1355	1208	498	2014
20	1930	1441	1206	507	2126	1945	1496	1297	526	2159

An important point to mention is that the times required for recommending the randomly selected hidden pages using the modified CF algorithms are similar to those of the recommendation times for the most significant hidden pages (see Tables 17 and 24).

Chapter 7

Semantic Inclusion to Weighted Session Similarity Measure

In the earlier chapters we have shown that along with URL-similarity, consideration of user interest in terms of duration and frequency of visiting a page leads to considerably improved results in the quality of usage profiles created. From the experimental results, it has been found that improvement in recommendation performance can be achieved while using our weighted similarity measure also in obtaining the nearest prototype and in the selection of nearest neighbors for recommendation purpose. While URL-similarity is assumed to relate to content similarity, inclusion of web page semantics would certainly be more desirable. In this chapter, we propose a modification to our weighted session similarity measure for incorporating web page concept similarity. We first discuss the concept hierarchy, a way to represent the semantic relationship among the web pages. After that, we explain our proposal for measuring concept similarity between pages. Then we present our modified formulation of the weighted session similarity measure to include concept similarity.

7.1 Concept Hierarchy

The main goal of the recommender systems based on the web usage mining is to recommend a list of pages to the user by matching his/her current browsing pattern with a set of usage

profiles. Inclusion of web usage mining in recommender systems overcomes some limitations of traditional personalization techniques, i.e., collaborative filtering. However, insufficient usage data is a major problem in the performance of usage-based recommendation systems. Further, addition of new pages in the website may often make the personalization systems base their recommendations on obsolete usage models.

Recently, a lot of research work has been done to integrate domain knowledge of the website into web usage mining in order to improve the performance of personalization system [1, 6, 37, 62]. Domain knowledge of the website is available from domain experts, website designers, or even from the web page contents. In the form of domain knowledge, a concept hierarchy, the website topology and/or semantic classification can be used. In order to achieve further enhancement in recommendation performance we propose to modify our weighted similarity measure by incorporating the semantics of web page content via a concept hierarchy. Generally, web pages with high url-similarity possess high conceptual similarity. However, low url-similar pages can also be conceptually related. For example, two web pages possessing information about two individual professors may have low url-similarity, but from conceptual point of view they possess high semantic relation while considering the concept, "teaching". Therefore, it is likely that the two users are semantically highly similar, while accessing these pages. This perception forms the primary motivation for our proposed semantic similarity measure among the web pages.

A concept hierarchy is an abstract hierarchical taxonomy, which is used to present the semantic relationship among the web pages [6, 41, 42]. In this hierarchy, each internal node represents a concept used to relate pages, while each leaf represents a web page. Typically, the concepts are used to characterized the web pages. Each edge in the hierarchy maintains an "IS-A" relationship between the nodes. A parent node possesses a general concept and abstracts the contents of all of its descendent nodes. The descendent nodes are comparatively less generalized. Therefore, there exists a general to specialized concept-ordering throughout the hierarchy, where the root node represents the most general concept and the leaves represents the most specialized concepts (i.e., pages). An important point to

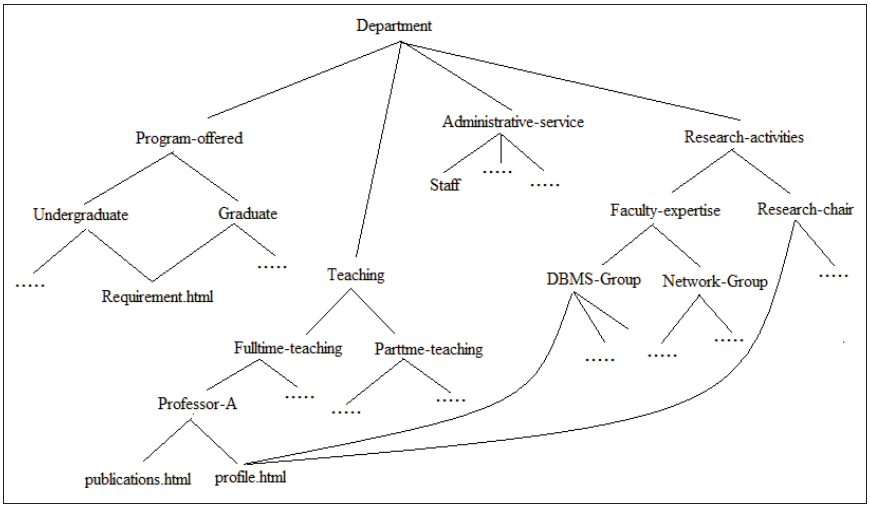


Figure 14: A Part of Concept Hierarchy of "CS Department" Website.

mention is that a web page can have more than one concept, i.e., characterizing the page from different points of view. For example, a professor-profile page can be characterized by an academician point of view. However, at the same time it can be characterized by an administrator point of view, if the professor is in an administrative position in the University. Therefore, it is reasonable to present the concept hierarchy of a website using a directed acyclic graph (DAG) [6, 46, 55]. Fig. 14 presents a part of concept hierarchy of a "CS Department" website of a University.

7.2 Proposed Semantic Similarity Measure (SESM) Between Web Pages

Generally, each usage session represents browsing intent of a user. In a concept hierarchy, web pages are organized based on their concepts, maintaining a general to specialized concept-ordering. Each page, presenting one or more specialized concepts, contains all the concepts related to all of its ancestors on the path(s) leading to the root from the page. Therefore, when a user accesses a page, it is reasonable to assume an indirect access to all it's related concepts in other nodes. While counting the number of accesses to a node,

we can easily deduce that the root, which is the ancestor of all nodes and contains all the concepts at the most general level, possesses the highest access-count. There exists a decreasing rate of the access-count from the root to the leaves in the entire hierarchy. The probability of node-wise access is based on this count. Clearly, the root contains the highest probability of 1.0 and others have their probability accordingly. In contrast, while ranking the nodes with respect to the specialized information, an increasing trend is observed from the ancestor nodes to the descendents, where the root having the most general concept is placed at the rear and the leaves possessing the most specialized information are placed at the front of the ranked-node list.

Based on these reverse properties of the probability of access-count (PAC) and the acquired specialized information (SPI) of a node, we derive an exponential relationship between them. Equation (20) defines the expontential correlation between PAC and SPI for a node N_i.

$$SPI_{N_i} = e^{-PAC_{N_i}} \tag{20}$$

The result obtained from equation (20) is used as a weight to the node N_i.

Since the concept hierarchy is essentially a directed acyclic graph (DAG), it is likely to have more than one path from a leaf node to the root. Let N_i and N_j denote two leaf nodes, which correspond to the pages url_i and url_j respectively. Let A_{N_i}=$\{w_1(a_{N_i})$, $w_2(a_{N_i})$, ..., $w_x(a_{N_i})\}$ denote the weights of all ancestors of N_i, where each ancestor belongs to one individual path and has the highest weight among all ancestors on the same path to the root (i.e., as far from the root as possible). Again, let A_{N_j}=$\{w_1(a_{N_j})$, $w_2(a_{N_j})$, ..., $w_y(a_{N_j})\}$ denote the weights of all ancestors of N_j, where each ancestor node is the highest weighted ancestor of each individual path. Let $A_{N_{ij}}$=$\{w_1(a_{N_{ij}})$, $w_2(a_{N_{ij}})$, ..., $w_z(a_{N_{ij}})\}$ denote the weights of all common ancestors of N_i and N_j, where each ancestor is attached to one distinct path, having highest weight among all common nodes on the same path. Our proposed semantic similarity measure between two nodes N_i and N_j (i.e., corresponding to the pages url_i and url_j respectively) is defined by equation (21).

$$Similarity_{sem}(N_i, N_j) = \frac{\max(\forall_z w_z(a_{N_{ij}}))}{\max(\max(\forall_x w_x(a_{N_i})), \max(\forall_y w_y(a_{N_j})))} \quad (21)$$

Our proposed semantic similarity measure satisfies the following properties:

(1) $0 < Similarity_{sem}(N_i, N_j) \leq 1$

(2) $Similarity_{sem}(N_i, N_j) = Similarity_{sem}(N_j, N_i)$

The following example illustrates the semantic similarity computation between the pages using equation (21).

Example 5.

We use the portion of concept hierarchy of "CS Department" website (see Fig. 14) for illustration. It is noted that each leaf corresponds to a web page of the website. We can calculate the access-count of all leaves directly from the weblog. Again, we can compute the access-count of other nodes from the access-count of their respective children.

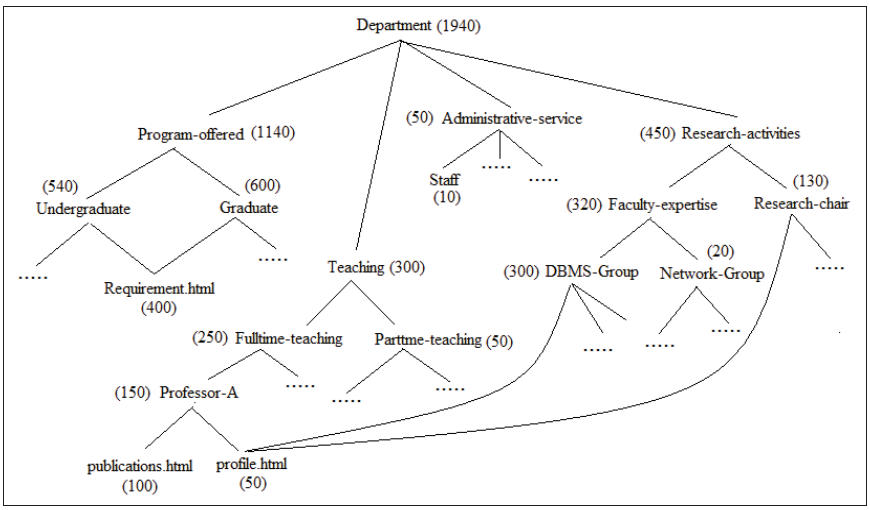


Figure 15: A Part of Concept Hierarchy of "CS Department" Website with Access-count.

Next, we compute the probability of access-count for each node, and lastly, using equation (20), we assign a weight to each node. Figs. 15, 16, and 17 show the example of weighting process to the concept hierarchy.

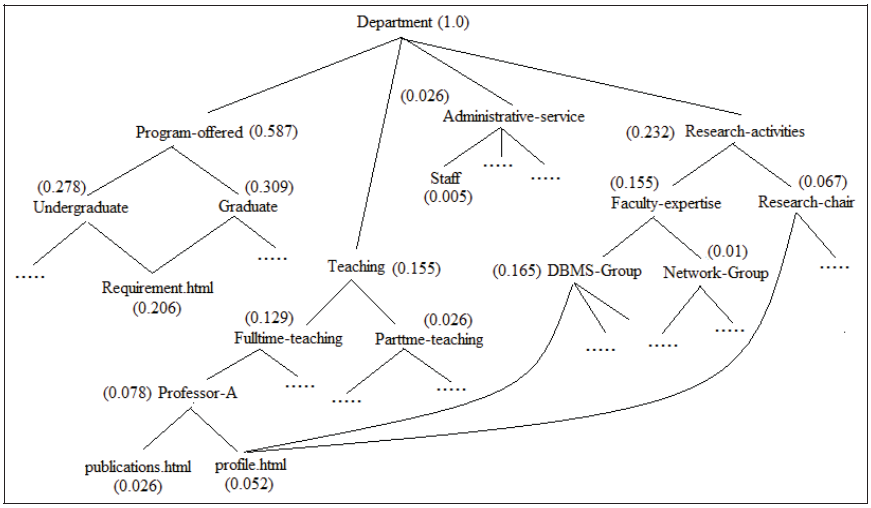


Figure 16: A Part of Concept Hierarchy of "CS Department" Website with Probability.

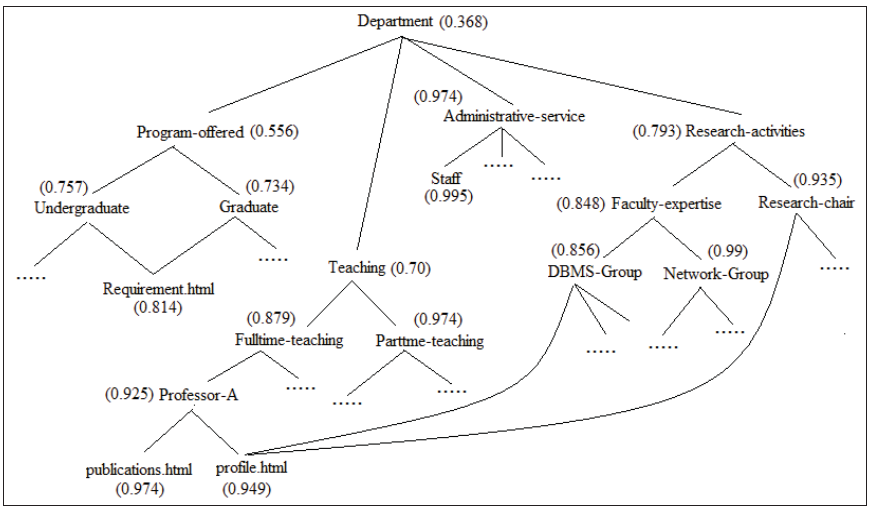


Figure 17: A Part of Concept Hierarchy of "CS Department" Website with Weight.

Let N_1, N_2, and N_3 denote three concepts in the concept hierarchy of "CS Department" Website (with respect to Fig. 14). Let N_1 corresponds to url_1="profile.html", N_2 corresponds to url_2="publication.html", and N_3 corresponds to url_3="requirement.html". Using equation (21), we compute $Similarity_{sem}(N_1,N_2)$=0.99 and $Similarity_{sem}(N_1,N_3)$=0.398 (with respect to Fig. 17).

7.3 Incorporation of SESM into Weighted Session Similarity Measure

We replace the URL-similarity among pages from our original weigthed session similarity measure (i.e., equations (11) to (13)) with the page semantic similarity. It has already been mentioned that URL-similarity among pages represents their topic (subject) similarity. Generally, a high URL-similar page pair indicates that they are conceptually related. However, low URL-similar pages can also be conceptually more related. Our weighted session similarity measure takes this aspect into consideration and gives higher similarity to those weighted sessions containing pages with low URL-similarity but high conceptual relation. We modify our weighted session similarity measure by making use of the session-wise page significance and page semantic similarity. Equations (24) to (23) define our new similarity measure, WSS_{sem}, for two weighted sessions ws_L and ws_K with $0<WSS_{sem}(\text{ws}_K,\text{ws}_L)\leq 1$.

$$WS_{sem_{1KL}} = \frac{\sum_{i=1}^{M} ws_K(Sig_{url_i}) \times ws_L(Sig_{url_i})}{\sqrt{\sum_{i=1}^{M} ws_K(Sig_{url_i})^2}\sqrt{\sum_{j=1}^{M} ws_L(Sig_{url_j})^2}} \tag{22}$$

$$WS_{sem_{2KL}} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{M} ws_K(Sig_{url_i}) \times ws_L(Sig_{url_j}) \times Similarity_{sem}(N_i, N_j)}{\sum_{i=1}^{M} ws_K(Sig_{url_i}) \times \sum_{j=1}^{M} ws_L(Sig_{url_j})} \tag{23}$$

$$WSS_{sem}(ws_K, ws_L) = \max(WS_{sem_{1KL}}, WS_{sem_{2KL}}) \tag{24}$$

Here, N_i denote a node of the concept hierarchy, which corresponds to url_i. Our modified weighted similarity measure $WSS_{sem}(\text{ws}_K,\text{ws}_L)$ ensures the following properties:

- Nonnegativity: $0<WSS_{sem}(\text{ws}_K,\text{ws}_L)\leq 1$.
- Identity: $WSS_{sem}(\text{ws}_K,\text{ws}_L) = 1$.
- Symmetry: $WSS_{sem}(\text{ws}_K,\text{ws}_L) = WSS_{sem}(\text{ws}_L,\text{ws}_K)$.
- Uniqueness: $WSS_{sem}(\text{ws}_K,\text{ws}_L) = 1$ means $\text{ws}_K=\text{ws}_L$.

In some cases, $WSS_{sem}(\text{ws}_K,\text{ws}_L)$ may violate Triangle Inequality:

- $WSS_{sem}(\text{ws}_K,\text{ws}_L) > WSS_{sem}(\text{ws}_K,\text{ws}_M) + WSS_{sem}(\text{ws}_M,\text{ws}_L)$.

Chapter 8

Conclusions and Future Work

Our research is in the domain of web recommender systems, which are part of the strategy in web personalization. The main task of recommender systems is to identify a set of items that will be of interest to individual users. This way they ease interactions of the users with the vast amount of information on the web by automatically suggesting "interesting" pages to the users.

Presently, collaborative filtering is the most successful approach for developing the recommender systems, and is extensively used in many commercial recommender systems. Traditional collaborative filtering approach discovers a set of like-minded users in real-time, that is those who have preferences similar to this active user, and makes use of the preferences of these like-minded users for prediction or recommendation to this user. However, the computational complexity of traditional collaborative approach rises linearly with the number of users, which makes this approach less scalable for a large website with millions of users. In addition, the recommendation accuracy of this approach may decline due to the sparse nature of the user-rating dataset. Typically, users of a large website will often access only a small part of the site pertaining to the information of relevance to that user. This results in a rather sparse user-rating dataset. The like-minded user discovery process utilizes this dataset and computes similarity based on the common items in users preferences. In general, the common items in user browsing could be relatively small, thereby

reducing the reliability of similarity results and the recommendation accuracy.

To deal with these limitations of traditional collaborative filtering, model-based collaborative filtering techniques have been developed which are based on aggregate user preference patterns rather than individual user preferences. A number of web usage mining techniques such as clustering, association-rule generation, and etc., have been used to generate models of user preference patterns. Typically, this is represented as a set of usage profiles by clustering the users based on similar interests. Each profile captures common interests of a group of users accessing the website. Thus, the quality of usage profiles is critical for the recommendation performance. A high quality set of profiles helps enhance the accuracy of recommendation. However, the quality of profiles relies on how effectively the users are grouped together based on their similar interests.

Effective user grouping by clustering of web usage data can lead to usage profiles that are representative of like-mind users. High quality usage profiles can help improve performance of recommender systems. Successful clustering, however, depends on how well user interests are captured and accommodated by the similarity measure that is used. Our research has addressed this specific problem. We have defined a weighted session similarity measure to assess usage session similarity by considering both page significance and URL structure similarity. The page significance helps signify the similarity between two users' browsing interest. Moreover, the URL structural similarity assists to discover the topic (subject) similarity in two users' browsing. Then two model-based CF algorithms have been adapted to make use of this measure to demonstrate the effectiveness of this new measure. Numerous experiments confirm that our similarity measure helps discover effective usage profiles from the large web log data. Our experiments include performance comparison with four other popular similarity measures in use in this domain. Our weighted session similarity measure distinctly outperforms other measures by providing recommendations of superior quality, overall.

We have further proposed a new approach for measuring the concept similarity among web pages based on the concept hierarchy, and modified our weighted similarity measure

by this concept similarity measure. Generally, web pages with high URL-similarity also possess high concept-similarity. However, low URL-similar pages does not automatically imply low concept-similarity as well. Developing a concept hierarchy for a large web site like our example of a university's web site with over 12,000 pages is an onerous task. It has to be done manually and by experts. Hence we could not carry out any experiments with this similarity measure.

In future, we aim to develop a fuzzy clustering technique to group the usage sessions based on their semantic similarity and to propose a recommender algorithm that exploits semantically annotated usage profiles for making the recommendation.

We have proposed our weighted similarity based on the set-based session representation as the page-access ordering is not needed to be preserved for the recommendation purpose. However, for the user navigation-path finding application, it is required to follow the page-access sequence order. We plan to extend our weighted similarity for the sequence-based session similarity computation in order to use this measure for any sequence-based web applications.

In addition, we would also like to incorporate weighted similarity measure in an item-based collaborative filtering system to deal with scalability problems in the user-based collaborative filtering systems.

Bibliography

[1] P. Achananuparp, H. Han, O. Nasraoui, and R. Johnson. Semantically enhanced user modeling. In *Proceedings of the 2007 ACM symposium on Applied computing (SAC'07)*, pages 1335–1339, 2007.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)*, pages 487–499, 1994.

[4] B. Berendt, B. Mobasher, M. Nakagawa, and M. Spiliopoulou. The impact of site structure and user environment on session reconstruction in web usage analysis. In *Proceedings of the WEBKDD'02 Workshop on Web Usage Analysis and User Profiling*, pages 159–179, 2002.

[5] B. Berendt, B. Mobasher, M. Spiliopoulou, and J. Wiltshire. Measuring the accuracy of sessionizers for web usage analysis. In *Proceedings of the Workshop on Web Mining at the First SIAM International Conference on Data Mining*, pages 7–14, 2001.

[6] A. Bose, K. Beemanapalli, J. Srivastava, and S. Sahar. Incorporating concept hierarchies into usage mining based recommendations. In *Proceedings of the WEBKDD'06 Workshop on Web Usage Analysis and User Profiling*, pages 110–126, 2006.

[7] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, 1998.

[8] G. Castellano, A. M. Fanelli, C. Mencar, and M. A. Torsello. Similarity-based fuzzy clustering for user profiling. In *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology Workshops (WI-IATW'07)*, pages 75–78, 2007.

[9] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. *Computer Networks and ISDN Systems*, 27(6):1065–1073, 1995.

[10] P. K. Chan. A non-invasive learning approach to building web user profiles. In *Proceedings of the WEBKDD'99 Workshop on Web Usage Analysis and User Profiling*, pages 7–12, 1999.

[11] M. S. Chen, J. S. Park, and P. S. Yu. Efficient data mining for path traversal patterns. *IEEE Transactions on Knowledge and Data Engineering*, 10(2):209–221, 1998.

[12] S. L. Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent Fuzzy Systems*, 2(3):267–278, 1994.

[13] R. Cooley, B. Mobasher, and J. Srinivastava. Web mining: Information and pattern discovery on the world wide web. In *Proceedings of International Conference on Tools with Artificial Intelligence*, pages 558–567, 1997.

[14] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.

[15] M. Eirinaki and M. Vazirgiannis. Web mining for web personalization. *ACM Transactions on Internet Technology*, 3(1):1–27, 2003.

[16] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: a constant time collaborative filtering algorithm. *Communications of the ACM*, 4(2):133–151, 2001.

[17] S. Gunduz and M. T. Ozsu. A web page prediction model based on clickstream tree representation of user behavior. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003)*, pages 535–540, 2003.

[18] D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1(1):49–75, 2000.

[19] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 230–237, 1999.

[20] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.

[21] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems (CHI'95)*, pages 194–201, 1995.

[22] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115, 2004.

[23] R. Ivncsy and S. Juhsz. Analysis of web user identification methods. In *Proceedings of the 4th International Conference on Computer, Electrical and System Science, and Engineering (CESSE 2007)*, pages 70–76, 2007.

[24] H. Keqin, H. Liang, and X. Weiwei. A new effective collaborative filtering algorithm based on user's interest partition. In *Proceedings of the 2008 International Symposium on Computer Science and Computational Technology (ISCSCT'08)*, pages 727–731, 2008.

[25] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, 1995.

[26] C. Li and Y. Lu. Similarity measurement of web sessions by sequence alignment. In *Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing Workshops (NPC'07)*, pages 716–720, 2007.

[27] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[28] H. Liu and V. Keselj. Combined mining of web server logs and web contents for classifying user navigation patterns and predicting users' future requests. *Data and Knowledge Engineering*, 61(2):304–330, 2007.

[29] U. Manber, A. Patel, and J. Robison. Experience with personalization of yahoo! *Communications of the ACM*, 43(8):35–39, 2000.

[30] M. J. Martin-Bautista, M. A. V. Miranda, and V. H. Escobar-Jeria. Obtaining user profiles via web usage mining. In *Proceedings of the 2008 IADIS European Conference on Data Mining*, pages 73–76, 2008.

[31] M. R. McLaughlin and J. L. Herlocker. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 04)*, pages 329–336, 2004.

[32] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, pages 187–192, 2002.

[33] K. Miyahara and M. J. Pazzani. Improvement of collaborative filtering with the simple bayesian classifier. *Computer and Information Science*, 43(11):1–28, 2002.

[34] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Communications of the ACM*, 43(8):142–151, 2000.

[35] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Effective web personalization based on association rule discovery from web usage data. In *Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM'01)*, pages 9–15, 2001.

[36] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery*, 6(1):61–82, 2002.

[37] B. Mobasher, X. Jin, and Y. Zhou. Semantically enhanced collaborative filtering on the web. In *Proceedings of the First European Web Mining Forum (EWMF'2003)*, pages 57–76, 2003.

[38] S. Nadi, M. H. Saraee, and A. Bagheri. A hybrid recommender system for dynamic web users. *International Journal of Multimedia and Image Processing*, 1(1):3–8, 2011.

[39] O. Nasraoui. World wide web personalization. In *Encyclopedia of Data Warehousing and Mining, J. Wang, (ed.), Idea Group*, pages 1235–1241, 2005.

[40] O. Nasraoui, H. Frigui, A. Joshi, and R. Krishnapuram. Mining web access logs using relational competitive fuzzy clustering. In *Proceedings of the Eight International Fuzzy Systems Association World Congress(IFSA'99)*, 1999.

[41] O. Nasraoui and E. Saka. Web usage mining in noisy and ambiguous environments: Exploring the role of concept hierarchies, compression, and robust user profiles. In *Proceedings of WebMine*, pages 82–101, 2006.

[42] S. Paulakis, C. Lampos, M. Eirinaki, and M. Vazirgiannis. Sewep: using site semantics and a taxonomy to enhance the web personalization process. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD03)*, pages 99–108, 2003.

[43] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: a hybrid memory- and model-based approach. In *Proceedings*

of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI00), pages 473–480, 2000.

[44] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI'01)*, pages 437–444, 2001.

[45] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.

[46] R. Samizadeh and B. Ghelichkhani. Use of semantic similarity and web usage mining to alleviate the drwabacks of user-based collaborative filtering recommender systems. *International Journal of Industrial Engineering and Production Research*, 21(3):137–146, 2010.

[47] M. K. Santhisree and D. A. Damodaram. Ssm-dbscan and ssm-optics: Incorporating new similarity measure for density based clustering of web usage data. *International Journal on Computer Science and Engineering*, 3(8):3071–3083, 2011.

[48] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'01)*, pages 285–295, 2001.

[49] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen. *Collaborative Filtering Recommender Systems*. The Adaptive Web: Methods and Strategies of Web Personalization. P. Brusilovsky, and A. Kobsa and W. Nejdl (ed.), Springer-Verlag, 2007.

[50] C. Shahabi, F. Banaei-Kashani, Y.-S. Chen, and D. McLeod. Yoda: An accurate and scalable web-based recommendation system. In *Proceedings of the Sixth International Conference on Cooperative Information Systems*, pages 418–432, 2001.

[51] C. Shahabi, A. M. Zarkesh, J. Adibi, and V. Shah. Knowledge discovery from users web-page navigation. In *Proceedings of the Seventh International Workshop on Research Issues in Data Engineering (RIDE'97) High Performance Database Management for Large-Scale Applications*, pages 20–29, 1997.

[52] G. Shani, D. Heckerman, and R. I. Brafman. An mdp-based recommender system. *Journal of Machine Learning Research*, 6(1):12651295, 2005.

[53] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating word of mouth. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, 1995.

[54] M. Spiliopoulou, B. Mobasher, B. Berendt, and M. Nakagawa. A framework for the evaluation of session reconstruction heuristics in web-usage analysis. *INFORMS Journal on Computing*, 15(2):171–190, 2003.

[55] R. Srikant and R. Agrawal. Mining generalized association mining rules. In *Proceedings of the 21th International Conference on Very Large Data Bases (VLDB'95)*, pages 407–419, 1995.

[56] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations*, 1(2):12–23, 2000.

[57] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–19, 2009.

[58] Z. Su, Q. Yang, and H. J. Zhang. A prediction system for multimedia pre-fetching in internet. In *Proceedings of the Eighth ACM International Conference on Multimedia*, pages 3–11, 2000.

[59] B. S. Suryavanshi, N. Shiri, and S. P. Mudur. An efficient technique for mining usage profiles using relatin fuzzy subtractive clustering. In *Proceedings of the 2005 International Workshop on Challenges in Web Information Retrieval and Integration (WIRI'05)*, pages 23–29, 2005.

[60] B. S. Suryavanshi, N. Shiri, and S. P. Mudur. A fuzzy hybrid collaborative filtering technique for web personalization. In *Proceedings of the 3rd Workshop on Challenges in Intelligent Techniques for Web Personalizationation (ITWP'05), in Conjunction with the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.

[61] S. Vucetic and Z. Obradovic. Collaborative filtering using a regression-based approach. *Knowledge and Information Systems*, 7(1):122, 2005.

[62] D. Vuljanic, L. Rovan, and M. Baranovic. Semantically enhanced web personalization approaches and techniques. In *Proceedings of the 32nd International Conference on Information Technology Interfaces (ITI'10)*, pages 217–222, 2010.

[63] S. Wang, C. Xu, and R. Wu. Cluster method based on fuzzy multisets for web pages and customer segments. In *Proceedings of the 2008 International Seminar on Business and Information Management (ISBIM'08)*, pages 125–128, 2008.

[64] J. Xiao and Y. Zhang. Clustering of web users using session-based similarity measures. In *Proceedings of the 2001 International Conference on Computer Networks and Mobile Computing (ICCNMC'01)*, pages 223–228, 2001.

[65] J. Xiao, Y. Zhang, X. Jia, and T. Li. Measuring similarity of interests for clustering web-users. In *Proceedings of the 12th Australasian Database Conference (ADC'01)*, pages 107–114, 2001.

[66] G. R. Xue, C. Lin, Q. Yang, W. Xi, H. J. Zeng, and Y. Yu. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*, pages 114–121, 2005.

[67] T. W. Yan, M. Jacobsen, H. Garcia-Molina, and U. Dayal. From user access patterns to dynamic hypertext linking. *Computer Networks and ISDN Systems*, 28(7-11):1007–1014, 1996.

[68] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel. Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):56–69, 2004.

[69] Y. X. Yu and X. W. Wang. Web usage mining based on fuzzy clustering. In *Proceedings of International Forum on Information Technology and Applications (IFITA'09)*, pages 268–271, 2009.

[70] J. Zhang and A. A. Ghorbani. The reconstruction of user sessions from a server log using improved time-oriented heuristics. In *Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04)*, pages 315–322, 2004.