A Novel Real-Time Browsing Assistance System Based on Web User Behaviors

by

Syed Tauhid Zuhori

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Software Engineering & Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Web traffic and e-commerce activities are increasing rapidly; hence, understanding the behavior of users based on their interactions with websites is becoming more and more important. To do so, web usage mining is needed. This research analyzes web clickstream data to extract usage patterns. There are two major challenges involved in Web usage mining. The first one is preprocessing the raw data to provide an accurate picture of how a website is used. The second one is to present the rules and patterns that are potentially interesting to the users by filtering the results. This forms the basis for this thesis, where a novel real-time system is discussed. This system builds personalized browsing assistance based on website user request(s) submitted to the web server and past user(s) behavior. Our proposed system is of crucial importance to users browsing the internet. Providing accurate link suggestions is one of the advantages of the system. This has been further developed to a live screenshot of the suggested web page. This enables the user to preview the content before making visiting the web page. Besides this, the proposed system can provide suggestions based on the user's browser and operating system. This means that every browser and operating system has a unique suggestion model customized to its user. To evaluate the system, we provide a user study, case studies and conduct experiments on five datasets to verify the effectiveness of our proposed system.

Preface

Chapter 3 of this thesis has been published as Syed Tauhid Zuhori and James Miller, "**Real-Time Browsing Assistant on Web**", IDAIS International Journal on WWW/Internet, Volume-16, Issue-2, Pages 1-18 and Chapter 2 has been accepted for publication as Syed Tauhid Zuhori and James Miller, "**An Automated Web Structure-Based Method for Predicting the Importance of a Webpage"**, London Journal of Engineering Research in Volume 22 Issue 2. For both journal, I was responsible for the analysis, simulation, and interpretation of results as well as the manuscript composition. James Miller was the supervisory author and was also involved with the concept formation, interpretation of results, and manuscript composition.

To

Jarina Islam
(My Beloved Mother)

Acknowledgment

I would like to express my sincere gratitude and appreciation to my advisor, Professor James Miller. I cannot thank you enough for your endless support over the past several years.

I would also like to acknowledge the financial support of IBM, Canada.

Finally, I would like to present my special thanks to my parents, and my siblings.

Thank you all!

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 PROBLEM STATEMET and RESERCH MOTIVATIONS

Our automated system helps users to find their optimal page to be visited next. A good illustration of a basic application that has implemented part of this technology is YouTube. Once videos on YouTube get a specified number of views then the account holder is paid. This transaction occurs where the account must be an AdSense version, the YouTube authority must review the channel, and the channel must have the necessary permissions for advertising. This information is included in the help section of the YouTube interface; see Figure 1.1. This figure indicates that a YouTube user, must locate this setting in the help section and proceed to navigate through settings to set up the account. This paper documents a novel real-time system where the steps in setting up such an account, for instance, will be automatically generated in the suggestion bar. This improves the user experience and improves on the functionality of the application.

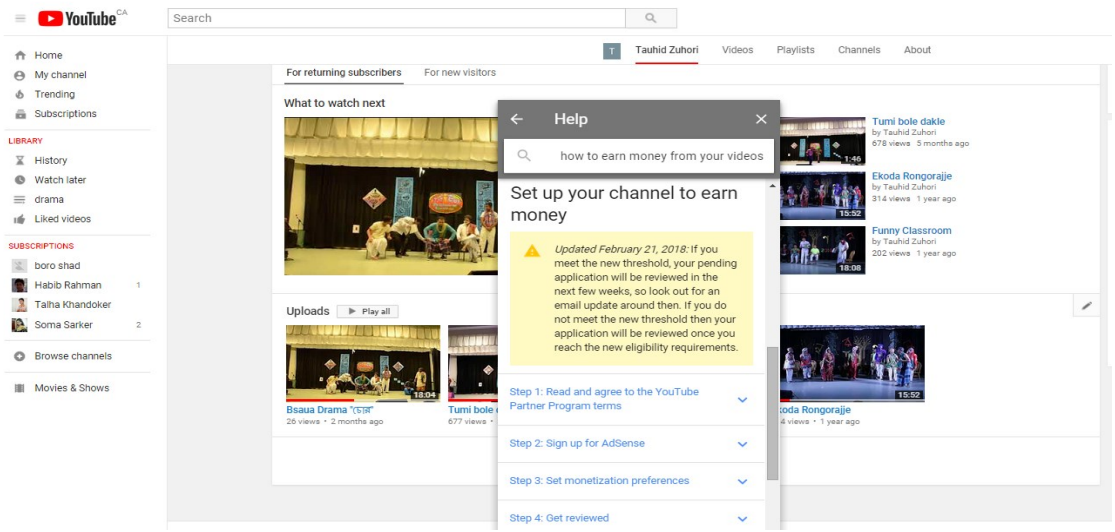


Figure 1. 1 Screenshot of YouTube "Account settings" page

Our proposed assistance system can save browsing time for web users. For instance, visa application processes are lengthy process. Recently part of the process has been digitized. A perfect example is an application for a temporary resident visa on the "Immigration and Citizenship"

website of Canada. The process involves logging into a website and building a profile by answering questions; Figure 1.2 illustrates that. The website combs through the answers and provides a result as to whether the user is eligible or not for a visa. This is followed by a process of uploading several filled-out forms. The user has to navigate through the website to find the section where they download and upload the forms once they are filled. Figure 1.3 illustrates this with a screenshot. However, this can be improved with the proposed system. The system will direct the new user based on different recorded (previous) user experiences. This improves the user experience by being more accessible and efficient.

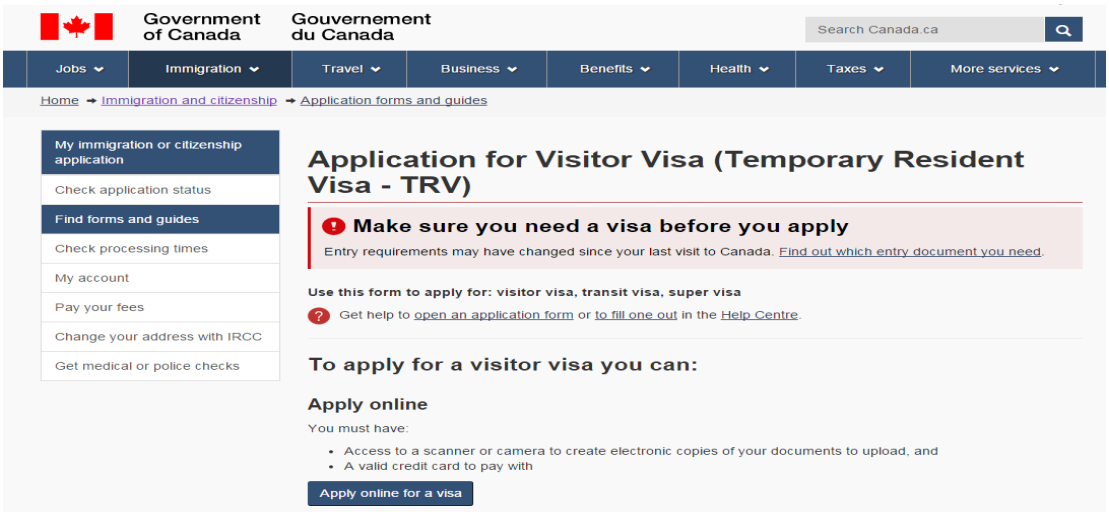


Figure 1. 2 Screenshot of CIC "Apply Online" page

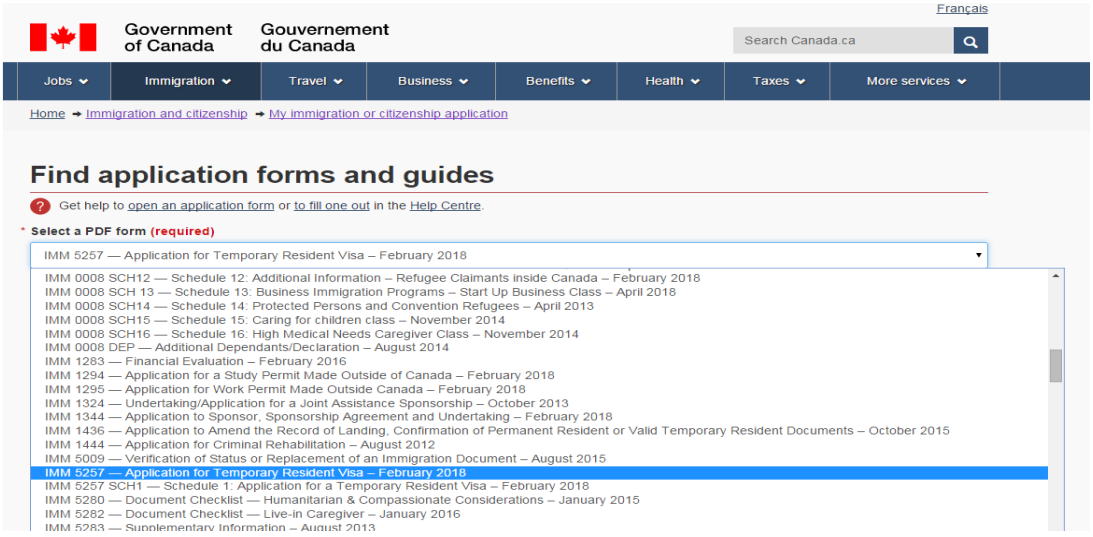


Figure 1. 3 Screenshot of CIC "Form Selection" page

1.2 RESEARCH CONTRIBUTIONS

An interactive system is developed that can interact, in real-time, with users. Our system will enable us to incrementally generate user-behavior models based on user-intensive web application browsing. Specifically, it takes the user navigation patterns as input data and generates an inference model of the website using a Discrete Time Markov Chain (DTMC) process that is continuously updated in real-time using a combination of Reinforcement learning (RL) and the Markov Decision processes (MDP). In the inference model, the nodes are the unique links of the website and the edges are the transition probabilities of moving between links. By analyzing the transition probabilities, we predict the users' appropriate links. This is finalized by building a real-time system that can generate suggestions by taking the user's requested link as input and providing appropriate suggestions. Our paper makes the following major contributions:

- A real-time suggestion generation system is constructed for websites that can be used as a plug-in to web browsers. If users find their expected link on the suggestion bar they can simply click and be redirected. This improves user experience since more accurate navigation data is presented.
- As a user searching for content online might take a lot of time, this has been eased in the proposed system by providing a screenshot of the suggested web links/ web pages. This gives the user a glimpse into the website before opening and exploring it.
- Different configurations of browsers and operating systems produce different visual results when painting the same HTML, CSS and JavaScript. Hence, we provide different guidance to a user based on their utilized browser and operating system. Such differentiation is unique to this paper.
- The paper evaluates two case studies, the "University of Alberta" and "RUET Online judge" websites, to demonstrate the effectiveness of our tool in improving user experience. These are real, live, mission-critical websites! While we process time-delay interaction patterns, the use of actual mission-critical web page data is exceedingly rare in the literature.
- As indicated above the system is based on browsers and operating systems and as such different versions of the same are used in the case studies. This includes three different

operating systems (Windows, Linux and MAC) and browsers (Chrome, Mozilla and Opera).

- A user study with two test cases is conducted in this study, one with our extension and a different one without extension. This is used in a practical setting where users provide feedback on the usability of the two approaches. This is then tabulated in statistical analysis to prove the proposed system's effectiveness with actual users.
- Finally, the results are evaluated and a cross-validation process is conducted to ensure the system produces the desired algorithmic results.

1.3 SUMMARY of OUTCOMES

We start our thesis by finding the clustering of the website users using their navigation patterns. Firstly, we clean the web server logs by using a traditional clustering approach. Then, we apply a Discrete Time Markov Chain approach to generate a model of the user behavior. For generating the nodes for the model, we use a technique (regular expressions) to find out the atomic propositions. Then we find a directed graph as an output of a DTMC inference process. Next, we apply spectral clustering on that directed graph, which works on the affinity of the graph nodes and divides the nodes into clusters. Finally, we use graph traversal algorithms and discover the navigation patterns of web users for each cluster. To evaluate the approach, we use server log files from the website www.ualberta.ca. We published the paper at the 17th International Conference on WWW/ Internet 2018 and the extended version in the Journal of Software Engineering & Intelligent Systems.

After that, we proposed another system that can also find the clustering of the users. The proposed system estimates "user similarity" by comparing an individual's historical usage patterns with the historical usage patterns of other known users, and subsequently forms of a graph to represent the entire set of user similarity metrics. Iterative Conductive Cut Clustering is used to partition the graph into clusters. We published the paper at the 17th International Conference on WWW/ Internet 2018 and the extended version is accepted in the Journal of Software Engineering & Intelligent Systems.

Next, we work on the importance value of the webpage. Our third project aims to develop a method to find the importance of web pages without using web browser data or invading the

privacy of users. Rather, it works on the structure of a website. To achieve this goal, we propose a novel method that can take webpage content as input and produce a score for each page automatically. Initially, we extract content from a web page in real-time. Subsequently, we consider two important factors based on the website structure: (1) "What is the minimum number of clicks needed to access web pages on a website?" and (2) "How a web page is linked with other web pages in a website?" We use a learning method to train our model by using the "web page views" results generated by "Google Analytics" and "SimilarWeb". Experiments and Case studies on the world's most popular websites show that our method can produce very effective results in real-time. We represent this project in Chapter 2. This paper is accepted in the London Journal of Engineering Research (Volume 22 Issue 2).

Finally, in our fourth project, we work on real-time data. This work builds a personalized browsing assistant based on the current user request submitted to a web server. The process involves developing a behavior model using a Discrete Time Markov Chain (DTMCs) inference process. This is then used to monitor user activities, and thereafter suggest "where to go next". Finally, it updates the model in real-time using a Markovian Decision Process (MDP). To evaluate the system, we provide a user study, case studies and conduct experiments on two datasets to verify the effectiveness of our proposed system. We published the paper at the 17th International Conference on WWW/ Internet 2018 and the extended version in IDAIS International Journal on WWW/Internet. We represent this journal in chapter 3.

Chapter 2: An Automated Web Structure-Based Method for Predicting the Importance of a Webpage

ABSTRACT

The aim of this chapter is to develop a method to find the importance of web pages without using web browser data or invading the privacy of users. Rather, it works on the structure of a website. To achieve this goal, we propose a novel method that can take webpage content as input and produce a score for each page automatically. Initially, we extract content from a web page in real-time. Subsequently, we consider two important factors based on the website structure: (1) "What is the minimum number of clicks needed to access web pages in a website?" and (2) "How a web page is linked with other web pages in a website?" We use a learning method to train our model by using the "web page views" results generated by "Google Analytics" and "SimilarWeb". Experiments and Case studies on the world's most popular websites show that our method can produce very effective results in real-time.

2.1 INTRODUCTION

The most noticeable developments in the Twenty-First century are the innovations that led to the Information Age. The Twenty-First century has all the characteristics of an Information Age as e-commerce takes center stage in our modern life. This is evident in the different enterprises that heavily depend on websites such as banking, shopping, education, hotelier services, and transport. Online shopping is probably one of the most successful innovations in e-commerce. Through online shopping, many startups have developed different franchises which depend on users' past buying history. This includes advertising, accessing additional customers through social media and marketing in general. Therefore, the primary target is to make a website more intuitive.

One of the most popular applications in this category is "Google Analytics" (www.analytics.google.com) In this case, the developer will review the most viewed web pages, a user's interest in a specific web page and the time spent on that specific web page. This process has been automated by "Google Analytics", perhaps the premier website analyzer in the marketplace. For "Google Analytics" to be practical, code has to be facilitated for the webserver,

which the admin uses to manage the analytics. Figure 2.1 shows the "Google Analytics" code segment that an admin has to set on their server to retrieve results. "Similarweb" (https://similarweb.com), a web mining application on website traffic, also analyzes the audience behavior of a website. However, it also uses the user's personal information.

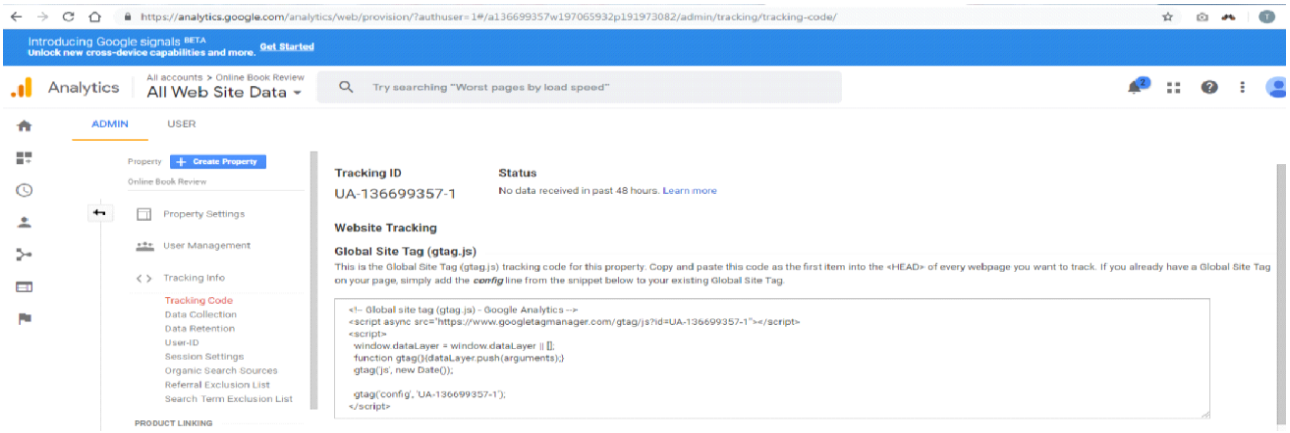


Figure 2. 1 Google Analytics Tracking code

Figure 2.2 shows that there is a message about using cookies displayed on their application. Therefore, the majority of the tools are using the user's personal information to determine the user's browsing behaviors. However, it is difficult to find the user's personal information or personal choices on websites. So a plausible, less intrusive, solution to this challenge is the use of a website's structure. Hence, we propose a system that tracks web pages in real-time and determines their importance by analyzing the structure of their website.

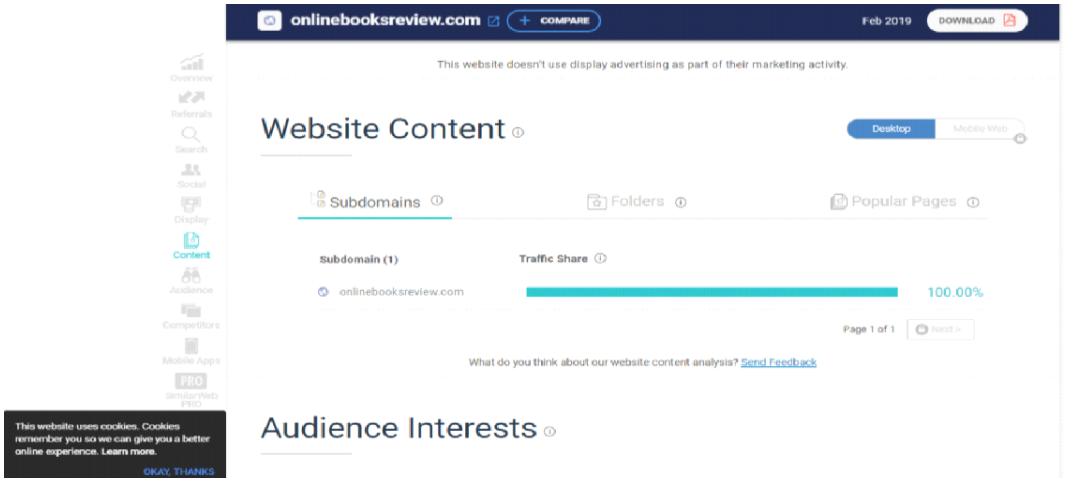


Figure 2. 2 Web application of "SimilarWeb"

To analyze the structure of web pages, we reviewed one hundred web pages. We selected these pages from the top twenty websites ranked by Alexa. By analyzing the structure, we find five important factors; i) The accessibility of the web pages, ii) The influence of a web page on a website, iii) The content of a web page, iv) Interacting web pages and v) Sharable web pages. The case study on the factors that influence the web pages' importance is represented in section 2. This work aims to provide a solution for online advertisements agencies, by providing an insight into the most viewed pages and providing suggestions to the web developer. This paper makes the following contributions;

- We develop an automated system that suggests areas that require improvement to make a particular web page more important. This is based on the structure of the website, and therefore no user data is required.
- By considering five different factors from the results of a Google Analytics case study on different websites, we propose a numeric measurement of the importance of web pages on a specific website and also represent the rank (Best, Good, Average and Poor) of the web page.
- We successfully conduct two case studies, by observing and analyzing the web pages of an "Online Book Review" website for twelve weeks, and conducting analysis on five hundred different websites from Alexa using the "Similar Web" tool, since we don't have server access to these sites.
- We conduct an additional case study on the web page "Contact Us" of the website "Online Book Review". We make four versions of it and show how this page can achieve more views by adopting our proposed system's suggestions.
- To validate our work, we use two types of validity – internal and external.
    - For internal validity, we represent the results in a confusion matrix. We automatically generate the features for the web pages using our extension that can extract the number of images, videos, links etc. Then we check the page manually and analyze the content of a web page. After that, we compare the results with manual results and produce a further confusion matrix.
    - For external validity, we also use both cases. We generate the features for both cases using our extension. Then we apply "CatBoost" to produce the importance

value and rank. In the case of the "Online Book Review" website, we use our generated features as input and the important value produced by "Google Analytics" as output. On the other hand, for websites ranked by Alexa, we also use the automatically generated features as input and the importance value generated by the "SimilarWeb" as output. Finally, in both cases, we use the "Pearson Correlation Coefficient" and "Spearman Correlation Coefficient" results to show the effectiveness of our work.

- Finally, we show four case studies on four types of rankings generated by our system with automatic suggestions. We manually check the effectiveness of our suggestions.

The rest of this chapter is organized as follows: in section 2.2 we represent our case studies for finding the important factors. Then in section 2.3, we review recent research on the topics of web mining. Section 2.4 describes the architecture of the proposed system for finding the importance of web pages. The experimental results are presented; and a discussion about the evaluation of these results, case studies, and validations are presented in Section 2.5. Finally, Section 2.6 summarizes the chapter and forms some conclusions.

2.2 FACTORS BEHIND THE WEB PAGES IMPORTANCE: CASE STUDIES

A case study is conducted to ascertain the factors behind the importance of a web page. The most popular websites from "Alexa" are selected for this study. Alexa describes each of the websites on their list based on the user's interest. Twenty websites are selected among the top list of websites for the case study. Some criteria are taken into consideration before choosing the websites for the study. Below we discuss which websites are excluded:

- "Google.com" is excluded because it is comprised of a search page where users need to type in their keywords. So, other website web pages depend on the user's keyword search. This made us exclude "Google.com" from our case study as we choose websites that are not dependent on any specific web page.
- Websites such as "Yahoo", "Facebook", and "Twitter" which require user accounts to access them are also excluded. The reason for its exclusion in our case study is that these sites can't be accessed as a guest.

- We ensure that all websites we work with have all the essential features such as images, texts, videos, and user interactions. So, YouTube is also excluded as most of its features include videos and hence this concentration is a single media type is considered problematic.
- We also ensure that a website written in English is selected for the case study. This is important as we feel that the website text is an important feature. So, we exclude websites that don't use the English language such as "baidu.com", "sohu.com", "Qq.com", and "Tmail.com". In reality, this rationale is simply to accommodate the limitations of the researchers.
- Pornographic websites are avoided because of their adult content.
- We also excluded one-page websites such as "thestartmagazine". The rationale behind this is that we feel that it's important to take into account the minimum number of clicks which won't be possible with a single-page website. Hence, the work presented in this paper only considers multiple-page websites as its domain of research.
- Websites like Wikipedia are also avoided because it is essentially a one-page website whereby any information clicked on appears on another Wikipedia web page which makes it difficult to measure web page hierarchy.

Hence, it is important to understand that our domain of application is limited to those websites which are not examples of our exclusion rules. We believe that the included sites are still the majority of websites (we use 500 websites from the first 656 websites from Alexa in this research). After developing the selection criteria, we spent a period of three months September 2021 to November 2021 monitoring suitable sites.

Table 2.1 shows the name and rank of the website that is selected from the "included list" for the case studies. The rank is recorded on a monthly basis (September 2021, October 2021 and November 2021) and changes over time; however, the rank is selected for the maximum number of days within the month. For example, if "amazon.com" was ranked 10 for 25 days in September, we choose that; a significant number of the websites are related to "e-commerce" in our case study.

Table 2. 1 Twenty websites selected from the top list of "Alexa"

Name of the Website	Ranking of the website according to "Alexa"		
	September 2021	**October 2021**	**November 2021**
Amazon.com	10	10	8
Blogspost.com	23	21	27
Microsoftonline.com	33	28	28
Ebay.com	41	45	37
Github.com	47	47	47
Imdb.com	48	48	48
office.com	50	52	55
stackoverlfow.com	51	49	49
Fandom.com	55	57	59
wordpress.com	57	56	52
imgur.com	58	60	60
Apple.com	61	61	61
Adobe.com	62	67	67
Amazon.in	65	65	69
Quora.com	79	81	78
Bbc.com	85	82	85
Roblox.com	90	95	96

Popads.com	91	93	93
Cnn.com	102	99	100
Spotify	107	120	120

Therefore, twenty websites are chosen from the "Alexa" top list for our case study. The selected websites were observed for three months. We chose 10 web pages all from a website for the case study. 5 out of these 10 pages are most visited while the other 5 represent the less visited pages. These data are collated from the "Similar Web" web application. Below are the steps we used for this study:

- The top 5 most visited and 5 less visited webpages names along with their number of views were extracted from the "Similar Web" app using an automated API (https://github.com/druidoff/similar-web-api/blob/master/SimilarWeb.php). The name of these web pages was collected for 3 months, between September and November 2021. There were variations to the most visited and less-visited pages daily. Based on our case study on 20 websites and 10 web pages each from selected websites, it implies that we collate data from 20 "Alexa" websites daily. This means within three months, we collected data from **18,200** web pages. After data collection from "Alexa", we proceeded to collect data for our case study from web pages we have earlier identified. We focus on the following critical features such as i) Web page contents, ii) Web page influence on a website, iii) Web page accessibility iv) web page interactions and v) sharable web pages.
- To collect web page accessibility data, a site map is created automatically. The technique used will be discussed in detail in section 2.4. An extension is created to generate the site map. We use this extension to all the 20 Websites **manually** and an XML sitemap (XML sitemap is a simple list of all the website pages) was produced. Since we cannot define the hierarchical structure in an XML sitemap, therefore, after we have generated the sitemap we will then use it to find the names of all websites' webpages. Also, several duplicate entries were observed in the sitemap, so after we have generated it automatically we then presented it manually in a hierarchical tree

structure. We took the names of the web pages from the sitemap and searched for them **manually** on the websites. After we have found out the webpage name through our search, we then put down the current web page name as "child" and "parent" for where the webpages were found. For example, if we found the “profile” page on the “Home” page that implies that we note the “profile” page as a child while the “home” page will be denoted as a parent. After we completed this pairing process, we were able to easily generate the website's tree structure. We were then able to discover the web page accessibility for all **18,200** web pages automatically for our case study. Table 2.2 shows the 2.3 monthly results of the accessibility of the web pages.

- After we found the tree structure of the websites, we then used the tree structure to generate a similarity graph of the websites. However, the graph is not sufficient enough to represent a website because of the high amount of edges appearing on the actual graph representation. For instance, let us assume we can access “Profile” web page from 3 separate web pages of the websites. On the tree structure, we set the “Profile” as a child of the “Home” page. So when a similarity graph is generated, only one edge will be shown while on the actual graph, 3 more edges are shown. To solve this issue, after we generated the similarity graph, we then automatically extracted the web pages name that can be visited through the current web page. After that, we then deleted the links that are not presented on the same websites (Suppose a link for sharing Facebook, a different website is found). We find these links **manually** also. Then, edges were set for all the web pages from the current in the similarity graph (Nodes represent web pages’ names in the similarity graph). We then find the web page's influence from that graph. We gave in-depth details in section 2.4. Table 2.3 shows the 3 months results of the accessibility of the web pages.
- We then collected the web page contents data (number of images, words, videos, weblinks), web page interactions (login, signup, checkout etc.), and shareable web pages (web pages capable of being shared to other social media websites) automatically. All these data were collected within 3 months.

Table 2. 2 Accessibility value of the web pages for both categories

Name of the Website	Accessibility of web pages					
	Category 1			Category 2		
	Max	Min	Median	Max	Min	Median
Amazon.com	3	2	2	1	0	1
Blogspost.com	2	1	2	1	0	1
Microsoftonline.com	3	2	2	2	0	1
Ebay.com	3	2	2	1	0	1
Github.com	2	1	1	1	0	1
Imdb.com	4	2	2	2	0	0
office.com	3	2	2	1	0	0
stackoverlfow.com	4	2	2	2	0	0
Fandom.com	2	1	1	1	1	0
wordpress.com	3	2	2	2	1	0
imgur.com	2	1	2	1	0	0
Apple.com	3	2	1	1	0	0
Adobe.com	2	1	2	1	1	0
Amazon.in	3	1	1	2	0	1
Quora.com	3	2	2	1	0	1
Bbc.com	4	2	3	1	0	1

Roblox.com	3	1	2	2	1	0
Popads.com	4	2	1	1	0	0
Cnn.com	3	1	2	1	1	0

Table 2. 3 Influence of a web page on both categories

Name of the Website	**Influence of a web page on a website**					
	Category 1			**Category 2**		
	Max	**Min**	**Median**	**Max**	**Min**	**Median**
Amazon.com	1.818	0.672	1.221	1.818	0.672	1.221
Blogspost.com	1.234	0.427	0.872	1.234	0.427	0.872
Microsoftonline.com	1.126	0.482	0.756	1.126	0.482	0.756
Ebay.com	1.112	0.426	0.728	1.112	0.426	0.728
Github.com	1.781	0.657	1.025	1.781	0.657	1.025
Imdb.com	1.289	0.429	0.821	1.289	0.429	0.821
office.com	2.114	0.782	1.412	2.114	0.782	1.412
stackoverlfow.com	1.782	0.678	1.129	1.782	0.678	1.129
Fandom.com	1.987	0.698	1.231	1.987	0.698	1.231
wordpress.com	2.112	0.772	1.467	2.112	0.772	1.467
imgur.com	1.123	0.419	0.758	1.123	0.419	0.758
Apple.com	1.256	0.425	0.857	1.256	0.425	0.857

Adobe.com	1.984	0.678	1.241	1.984	0.678	1.241
Amazon.in	2.123	0.782	1.435	2.123	0.782	1.435
Quora.com	1.678	0.578	1.098	1.678	0.578	1.098
Bbc.com	1.876	0.612	1.287	1.876	0.612	1.287
Roblox.com	1.987	0.682	1.257	1.987	0.682	1.257
Popads.com	1.876	0.662	1.298	1.876	0.662	1.298
Cnn.com	1.276	0.452	0.872	1.276	0.452	0.872

Table 2. 4 Number of images of web pages for both categories

Name of the Website	**Images on a webpage**					
	Category 1			**Category 2**		
	Max	**Min**	**Median**	**Max**	**Min**	**Median**
Amazon.com	39	7	17	13	2	6
Blogspost.com	10	2	7	5	0	4
Microsoftonline.com	11	3	6	4	1	2
Ebay.com	32	6	11	16	2	6
Github.com	4	0	2	2	0	1
Imdb.com	12	3	6	6	2	2
office.com	15	3	5	5	3	1
stackoverlfow.com	6	1	3	3	2	1

Fandom.com	11	2	5	4	2	2
wordpress.com	10	2	4	5	2	1
imgur.com	13	1	9	5	0	3
Apple.com	21	3	17	11	1	9
Adobe.com	12	2	10	4	1	4
Amazon.in	32	5	24	16	2	12
Quora.com	8	1	6	3	1	2
Bbc.com	12	2	5	6	1	2
Roblox.com	17	2	9	6	1	3
Popads.com	12	1	7	6	1	4
Cnn.com	31	3	17	11	2	1

Table 2. 5 Number of words of web pages for both categories

Name of the Website	Number of words in a web page					
	Category 1			Category 2		
	Max	Min	Median	Max	Min	Median
Amazon.com	265	102	221	67	17	51
Blogspost.com	900	321	567	78	21	55
Microsoftonline.com	121	81	109	189	101	156

Ebay.com	123	89	98	23	19	20
Github.com	289	218	265	28	12	22
Imdb.com	247	127	187	34	14	29
office.com	265	123	210	93	29	56
stackoverlfow.com	1081	657	891	244	128	182
Fandom.com	230	124	189	332	159	218
wordpress.com	429	128	321	134	29	98
imgur.com	129	98	121	321	129	228
Apple.com	287	129	189	453	239	329
Adobe.com	127	80	102	32	12	21
Amazon.in	328	213	289	24	15	18
Quora.com	821	578	682	87	12	56
Bbc.com	928	456	781	33	18	25
Roblox.com	278	189	221	29	21	27
Popads.com	210	178	192	21	11	18
Cnn.com	316	135	219	87	28	67

Table 2. 6 Number of interactions of web pages for both categories

Name of the Website	User Interaction on a web page					
	Category 1			Category 2		
	Max	Min	Median	Max	Min	Median
Amazon.com	33	3	23	3	0	2
Blogspost.com	23	2	17	3	0	2
Microsoftonline.com	32	4	15	6	0	2
Ebay.com	38	2	18	3	0	1
Github.com	24	1	16	3	0	1
Imdb.com	41	2	26	6	0	2
office.com	38	2	18	3	0	2
stackoverlfow.com	42	2	27	6	0	2
Fandom.com	28	1	21	3	1	1
wordpress.com	39	2	21	6	1	2
imgur.com	21	1	12	3	0	2
Apple.com	38	2	18	3	0	2
Adobe.com	27	1	16	3	1	1
Amazon.in	32	1	17	6	0	1
Quora.com	31	2	15	3	0	1
Bbc.com	49	2	28	3	0	1
Roblox.com	36	1	17	6	1	3

Popads.com	42	2	18	3	0	2
Cnn.com	32	3	17	3	1	1

Table 2. 7 Number of sharable web pages the web pages for both categories

Name of the Website	**Sharable web pages**					
	Category 1			**Category 2**		
	Max	**Min**	**Median**	**Max**	**Min**	**Median**
Amazon.com	6	4	4	2	0	2
Blogspost.com	4	2	4	2	0	2
Microsoftonline.com	6	4	4	4	0	2
Ebay.com	6	4	4	2	0	2
Github.com	4	2	2	2	0	2
Imdb.com	8	4	4	4	0	0
office.com	6	4	4	2	0	0
stackoverlfow.com	8	4	4	4	0	0
Fandom.com	4	2	2	2	2	2
wordpress.com	6	4	4	4	2	2
imgur.com	4	2	4	2	1	1
Apple.com	6	4	2	2	0	0

Adobe.com	4	2	4	2	2	2
Amazon.in	6	2	2	4	0	2
Quora.com	6	4	4	2	0	2
Bbc.com	8	4	6	2	0	2
Roblox.com	6	2	4	4	1	2
Popads.com	8	4	2	2	1	1
Cnn.com	6	2	4	2	1	1

The overall result is represented in Tables 2.8 – Table 2.10. We show a total of the six features here while we further show the number of times category 1 (5 most visited pages) exceeds category 2 (5 less visited pages) for the twenty websites and six features. The results are shown for three months. We can see that category 1 leads over category 2 in all features.

Table 2. 8 Number of times category 1 exceeds category 2 or vice versa (in case of maximum value)

Feature	The maximum value of both categories in three months					
	Sep 2021		Oct 2021		Nov 2021	
	Category 1	Category 2	Category 1	Category 2	Category 1	Category 2
Accessibility of the web pages	20	0	20	0	20	0
Influence of a web page in a web site	19	1	20	0	20	0

Images of the webpage	15	5	16	4	18	2
Texts of the webpage	16	4	16	4	17	3
User interactions of the web pages	18	2	17	3	17	3
Sharable web pages	19	1	17	3	19	1

Table 2. 9 Number of times category 1 exceeds category 2 or vice versa (in case of minimum value)

Feature	The minimum value of both categories in three months period					
	Sep 2021		Oct 2021		Nov 2021	
	Category 1	Category 2	Category 1	Category 2	Category 1	Category 2
Accessibility of the web pages	20	0	20	0	20	0
Influence of a web page in a web site	19	1	20	0	20	0
Images of the webpage	15	5	16	4	18	2
Texts of the webpage	16	4	16	4	17	3
User interactions of the web pages	18	2	17	3	17	3

Sharable web pages	19	1	17	3	19	1

Table 2. 10 Number of times category 1 exceeds category 2 or vice versa (in case of median value)

Feature	The median value of both categories in three months period					
	Sep 2021		Oct 2021		Nov 2021	
	Category 1	Category 2	Category 1	Category 2	Category 1	Category 2
Accessibility of the web pages	20	0	20	0	20	0
Influence of a web page in a web site	19	1	20	0	20	0
Images of the webpage	15	5	16	4	18	2
Texts of the webpage	16	4	16	4	17	3
User interactions of the web pages	18	2	17	3	17	3
Sharable web pages	19	1	17	3	19	1

After the case study and analysis of the results, we find the proposed factors the can influence the web pages' importance value:

- **Accessibility of the web pages:** This is based on the landing page of the website and the tabs available on the site. The landing page is technically the home page. Accessibility in this case, therefore, means that all the web pages that are accessible by one click from the landing page or the home page are more accessible than the pages that are accessible by two or more clicks from the home page.
- **Influence of a web page on a website:** Besides the accessibility of the web pages, we also observe that the web page that has more links and that can be accessed from more web pages has more influence on web users. Because the user comes to that page to visit the related web pages. In that case, the page views will be increased.
- **Content of the Web pages:** Text or information, images, and videos are referred to as content of the web pages. The page that contains more of this is considered more important.
- **Interacting Web pages:** Interactive web pages which require user input and show different outputs for different users by utilizing their inputs are as well considered to be more important.
- **Shareable Web pages:** This is in association with social media platforms, where some web pages contain links to social media platforms and their usability is clearly stated.

2.3 RELATED WORK

We examine the research work related to our study in this section; our work is related to web mining, which can be categorized into three active research areas depending on what components of web data are mined. The first one is Content Mining which is the process of extracting relevant information from the content of websites. The next one is Structure Mining which uses links and references within web pages. After analyzing that, It can obtain the underlying topology of the interconnections between web objects. The final one is usage mining which studies user access information from log server data. Our paper is based on website structure mining. However, our research uses structure mining to predict user behaviors. Hence, we include research work related to both mining topics.

Multazim et al. [2015] analyze whether classified ads can increase search engine rankings and increase the number of visitors to a website. They note that “Firefox” and “Google Chrome” are the most popular search engines. Hence, their study is based on ad’s data generated by those

search engines. They point out that posting and advertising are carried out by various auto-submit programs. They concluded that the installation of classified ad with 'Auto submit' increases the number of visitors.

Verma et al. [2015] make it clear that it is prudent for every organization to have a good website. Nonetheless, e-commerce is still in the developing stages in some countries such as India. They postulate that the challenging and dynamic needs of consumers are not satisfied in such countries where e-commerce is not well established. They make arguments based on research work that focuses on the design of a page ranking algorithm (SNEC). They explain that SNEC aids customers to search and compare products before purchasing. Finally, they recommend that business organizations need to structure their e-commerce websites to be more effective and usable.

Gleich et al. [2015] describe Google's PageRank method which evaluates the importance of web pages through their link structure. . They explain the process involved in determining the importance of web pages through various illustrations and mathematical formulae.

Khan et al. [2017] propose a new model, the popularity and productivity model (PPM). The model is based on a modular approach to finding the most influential bloggers. They describe in-depth the roles of the model's existing features and evaluate the proposed model by using data from real-world blogs. , they validate that PMM identifies influential bloggers. They make use of performance evaluation measures for the comparative analysis.

Tamimi et al. [2015] present the results of an experiment in which participants view fictitious web pages. They postulate various conceptual methods that are involved. Their study indicates that star reviews and familiarity with e-tailor (e-Bay or Amazon) are the main attributes that influence an individual's likelihood of purchasing products online. They further claim that their results are consistent with findings of previous research (Kim et al. [2010], Stocks et al. [2011]). They point out that while they encountered various limitations, their research can help to provide a more realistic task for a better comprehension of the attributes that have implications on consumers' decisions concerning the purchase of products online.

Zhen et al. [2016] combine the h-index and the PageRank algorithm. Their main aim is to find out the impact value of a publication. They construct the resulting PR-index for any

publication by evaluating the popularity of the source as well as the source publication authority. Therefore they propose their method should be added to technical indices.

Fatehu et al. [2016] propose a two-stage supervised approach to suggest news articles to users for a given state of Wikipedia. Initially, they suggest news articles to Wikipedia entities (article-entity suggestions) relying on a rich set of features. Then they determine the exact section in the entity page for the input article (article-section placement) guided by class-based section templates. They perform an evaluation of their approach based on ground-truth data that is extracted from external references in Wikipedia.

Zhen et al. [2017] observe that while there are many hypertext links on the web, only a few are clicked regularly. Based on this observation, they make use of mixed-effects hurdle models supplemented with descriptive insights and find out user preferences involved in clicking links on the web. They adopt the PageRank algorithm in their study. They utilize o large-scale data sets from Wikipedia (English version only) for their experiment. They conclude that Wikipedia users have a preference for navigating to articles that are in the periphery of the Wikipedia link network, compared to semantically similar articles, and to articles that are linked at the top of the left-hand side of the source article.

Thomas et al. [2019] research work is highly related to our research work. A research model was created with the use of a stimulus-organism-response (S-O-R) model (S. W. Khun et al. [2018]) to explicate how the social commerce features affect the website attention (stickiness) through ideas about cognitive and emotional factors. The meaning of the word “Website stickiness” entails the amount of attention received by a website from its users. E-commerce websites will find this very useful to their operations. Originating from environmental psychology, the S-O-R model postulates that certain stimulus affects the cognitive and emotional states of an individual; this then informs the individual’s response or behavior. Based on the S-O-R model, the cognitive and emotional states of the individuals facilitate a stimulus and response relationship. In the field of e-commerce, the S-O-R model has been widely tested by several studies to note how particular web features like stimuli (e.g., pictures, product descriptions, navigation aids) can influence consumers’ responses like buying behavior. Their research model is assessed in a controlled online experiment with 164 participants using e-commerce website variants with different social commerce feature richness levels. It was indicated in their results that cognitive

and affective dynamics affect feature richness positively, thereby increasing a website's stickiness. The result further concludes that e-commerce websites can be enhanced with a combination of functionally varied social commerce features. Unfortunately, they only work with high-level (abstract) issues of the website such as; user satisfaction, the usefulness of the websites, how users trust to share their data on websites etc. For validating this they design four versions of a website and take user responses on these issues. The high-level issues are varied from user to user. They take responses from a total of 212 participants and use the responses of 164 participants in their work (as 164 participants give an acceptable response) but it is still a very low number of users. Ultimately, this work fails to produce any guidance which would be meaningful to web designers or programmers. This is a key objective of our work.

To review the above-related work, we observe that, for finding the importance of web pages the previous research works on user navigation patterns, cookie information or other private data. Therefore, there are two significant problems with the previous research in this field. The first one is to collect user data. The second one is the use of previous data to find the solutions based on past user behaviors. So, most of the previous research work considers an old dataset of the website. For instance, suppose a new web page called "Donate Now" is included in a website for any incident. At that time that web page may attract more visitors but as the model learns from the previous dataset where the "Donate Now" link is not available, it would not be shown as the most visited page. So the previous works fail to decide in real-time and algorithms cannot quickly adapt to maintenance changes on an ongoing basis. In our work, we design a model according to the website structure. So, our work can give **real-time predictions without using any private data of users and automatically adapts to maintenance changes**. To our best knowledge, this is the first-ever work that can analyze the effectiveness of a web page by only analyzing the structure of the web pages.

2.4 METHODOLOGY

We have constructed an extension for web browsers. For this, we access the web pages' content and find the features by analyzing that content and page URL. These features are the input of our proposed system. Then for each web page, we find the importance value by using the page view results that were generated by "Google Analytics" and "SimilarWeb". We use this page view results to train our model and use the CatBoost Machine Learning (Liudmila et. al. [2010]) system

to produce the final importance metric. Finally, based on the metric, we rank the web pages on the scale of "Best," "Good," "Average" and "Poor." It is hoped that this procedure is straightforward enough, to make it accessible to most web site designers without requiring them to learn additional technology.

2.4.1 Feature Extraction

The information extracted from web pages and used as features are: i) what is the minimum number of clicks to visit those pages from the Homepage? ii) What is the number of images, texts, videos, links and scripts of the web pages? iii) How is a web page connected with other web pages on the website? iv) Are there any interactions with the users on the web pages? The establishment of these features is based upon the case study presented in Section 2. In this section, we discuss our methodology to extract these features from the web pages.

2.4.1.1 What is the minimum number of clicks to visit a web page

For finding the minimum number of clicks, we first extract the site map of the website using the method used by Brawer et al. [2017]. Starting with the "home" page, we get the all links that can be accessible and save them on the site map. Then we prune all the duplicate entries and increase the value of a minimum number of clicks. After that, we repeat these steps until there is no child found in the DOM Tree. Finally, when there is no child in the DOM tree, we find the site map of the websites with the minimum number of clicks. In our proposed method we use the ease of web pages' accessibility from a website. Therefore, after finding the minimum number of clicks, we find the easiness of the accessibility value of web pages (E) using; E = D − C ,Where D is the Maximum depth of a Tree and C is the minimum number of clicks.

2.4.1.2 What is the number of images, texts, videos, links and scripts on the web pages

We extract the DOM structure of a page and identify a summary of the content: (1) the number of images, (2) the number of videos, (3) the number of links on

the web page, (4) the amount of text (in words) and (5) the number of scripts on that page.

2.4.1.3 How is a web page connected with other web pages on the website

We produce an undirected graph for the web pages which represents the connectivity of all the pages on a website. Using this graph, we calculate the Eigen vector centrality for each of the nodes. (Here nodes mean the URLs of the website.) We use the adjacency matrix to find the Eigen vector centrality. For any vertex, v the relative Eigen Vector Centrality x can be defined as:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \, \epsilon \, G} a_{v,t} \, x_t$$

Where, $a_{v,t}$ is the adjacency matrix ($a_{v,t} = 1$, if there is an edge between the vertex v and t), M(v) is the set of neighbors of vertex, v and λ is a constant.

2.4.1.4 Are there any interactions with the users on the web pages

We loop through all the "<a>", "<nav>", "<submit>", "<form>" elements of the page. Before that, we collect keywords that are used for "Login," Signup," "Share on Facebook," Share on Tweeter," "Share on Google Plus," "Checkout". We collect these keywords by analyzing the Alexa top 400 websites. For this analysis, we only consider the home page of each website (Table 2.11). If we find these keywords within the tag elements, we infer that there is an interaction with the users.

Table 2. 11 Keywords collected from the Alexa top 400 websites

Keyword	Frequency	Keyword	Frequency	Keyword	Frequency
Log In	82	submit	24	Share on Facebook	15

Logon	17	Login	102	Share on Tweeter	12
Log	11	tweet	18	Share	45
Sign In	82	Facebook	28	Login Scope	18
Signin	27	googleplus	15	Share on Google	14
Signup	9	checkout	6	join	32
Sign up	31	check out	17	register	56

2.4.2 CatBoost learning to rank the web pages

Dealing with categorical features efficiently is one of the biggest challenges in machine learning. The most widely used technique to deal with categorical predictors is *one-hot-encoding*. The original feature is removed and a new binary variable is added for each category. Another way of dealing with categorical features is to use the so-called label-encoding technique that converts discrete categories into numerical features. Beyond these approaches, CatBoost (Liudmila et. al. [2010]) is a specialized version of Gradient Boosting Decision Trees (GBDT), which solves problems with ordered features while also supporting categorical features. It uses a technique where the trees included in the model are not independent but sequential. In other words, each predictor learns and improves from the mistakes and errors of the previous tree. In the end, all of the trees or predictors are combined to form the model but with non-uniform weights. Each tree is constructed by the following steps: 1) splitting calculations, ii) transformation of categorical data to numerical data, iii) construction of the tree, and, iv) computation of the values in the leaf nodes.

After the first split is selected on the tree, the same step is repeated for the next split only with a condition of ‘given the first split’. The same step is repeated with a similar condition until the whole tree is constructed. The model constructed includes a tree whose leaf values provide a

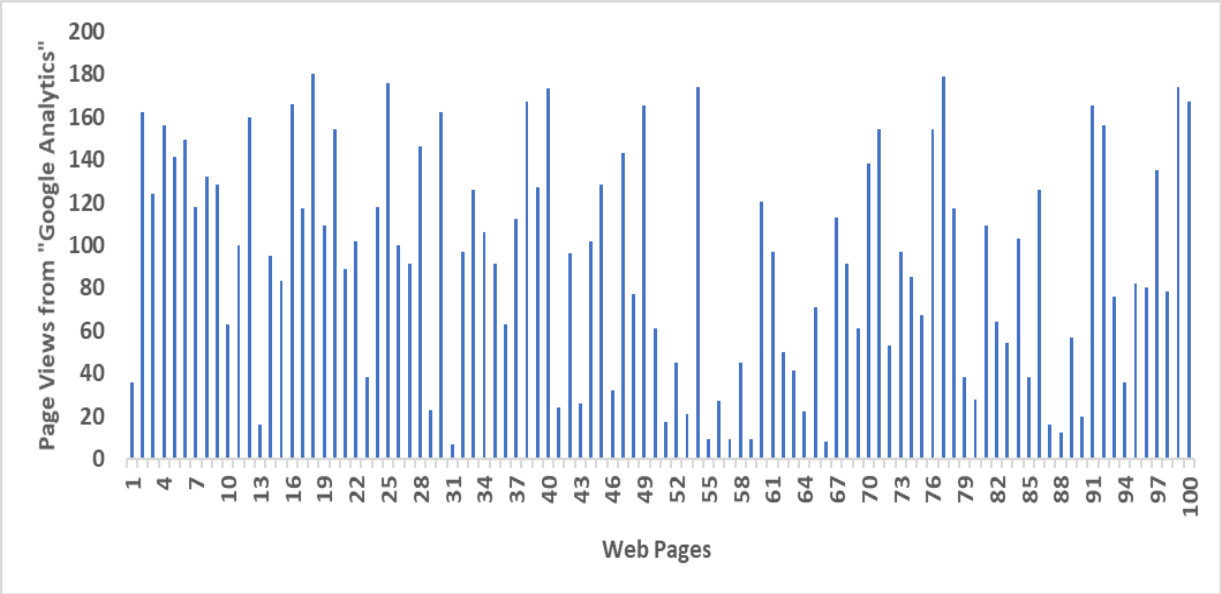
score which is our output —**importance values**. The score is further taken as input to classify itself as a rank. To categorize these values as one of the ranks, we break the range (output range from importance value) into four subranges (to decide what category a specific result falls under). (Using the ElbowMethod (Trupti et al. [2013]) estimates that k = 4 is the optimal partitioning). Further, to explain the subranges and the categories, the range in which the page is most likely to attract an audience and publicize or perform best is categorized as BEST, the one that draws a little less attention is named GOOD, the one that occasionally gets views is AVERAGE and the one that get rare or no audience at all is categorized as POOR.

2.5 RESULTS & EVALUATION

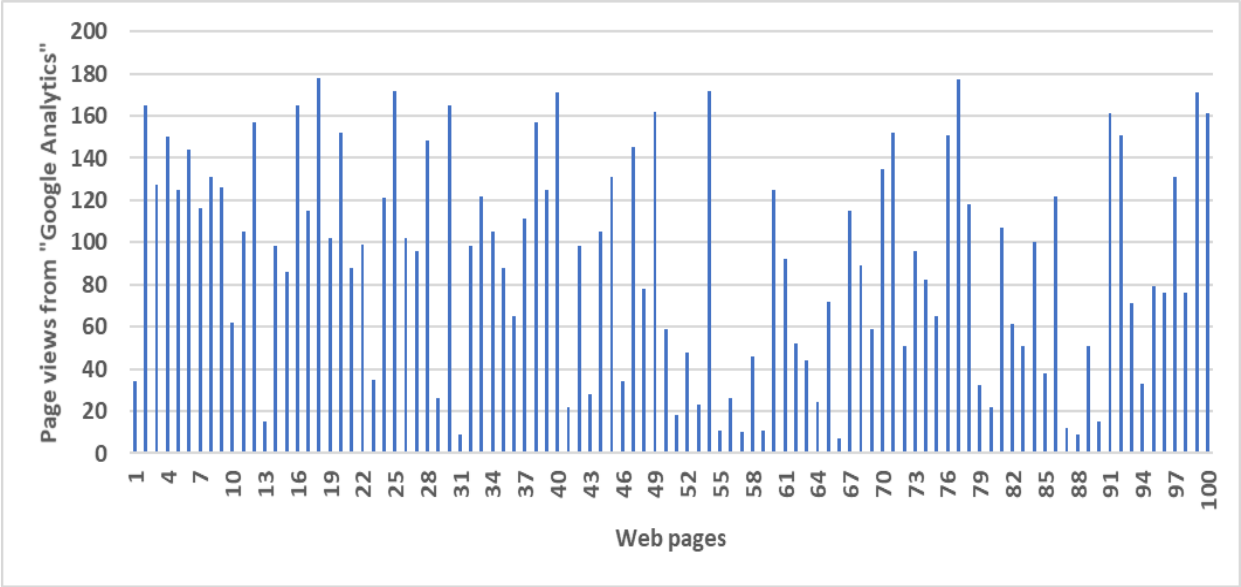
For evaluating our results, we generate two sorts of datasets. The first one has been generated from the "Google Analytics" results for the "Online Book Review" website (https://www.onlinebooksreview.com/). We obtain server access to this website for twelve weeks. The study follows best practices in maintaining users' anonymity and privacy, we obtained access between September 2021 and November 2021. We used the first eight weeks' data as the training set; and the last four weeks' data as test data. Besides this, we also generate a second dataset using "SimilarWeb." We choose Five Hundred websites from "Alexa" and generate datasets using the "SimilarWeb." We use the first four hundred websites as the training dataset and the rest One Hundred as the testing dataset. Then we show the results, and the importance values generated by our system. After that, we represent three case studies; at first one we represent how our system can extract the features from the web page, we represent four different pages for which our system generates the four different scores; "Poor", "Average", "Good" and "Best" according to their value and importance, in last one we show a case study on the web page "Contact Us"(https://www.onlinebooksreview.com/contact) of the web site "Online Book Review". We make four versions of it and show how this page can achieve more views by adopting our proposed system's suggestions. Finally, we represent two types of validity experiments to prove the effectiveness of our system; Internal and external.

2.5.1 Dataset Visualization

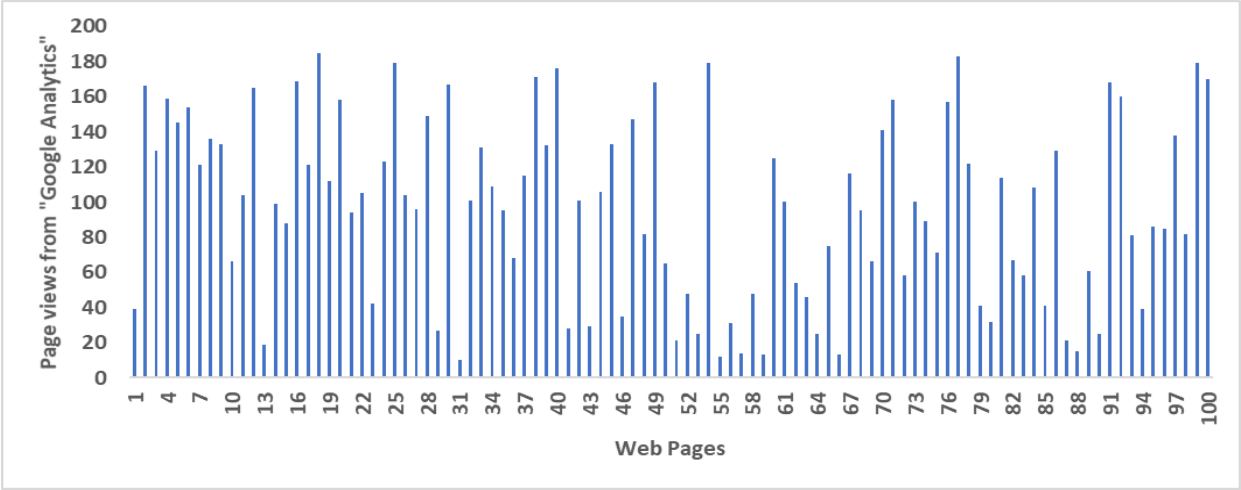
We use a total of 8 datasets in our research. We define the datasets as “Dataset 1” to “Dataset 8”. The first 3 datasets (Datasets 1 to 3) are based on the web pages of the website “Online Book Review”. Dataset 1 contains the data for September 2021; Dataset 2 contains the data for October 2021, and Dataset 3 contains the data for November 2021. There are a total of 239 web pages on the “Online Book Review” website; however, we select 100 web pages from them. We discard pages that are similar (such as “Articles on Programming”). In that case, we select one web page from each group. The combination of Datasets 1 and 2 is used as the training set, and Dataset 3 is used as testing. Figure 2.3 shows the “Number of views” results from the “Google Analytics” for each Dataset.



Dataset 1



Dataset 2

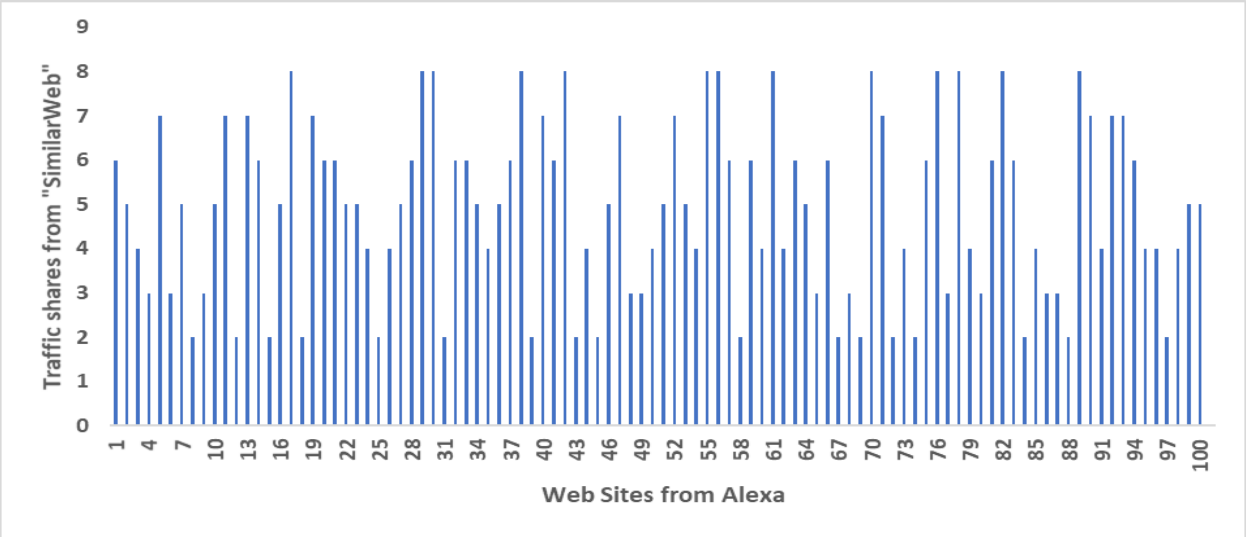


Dataset 3

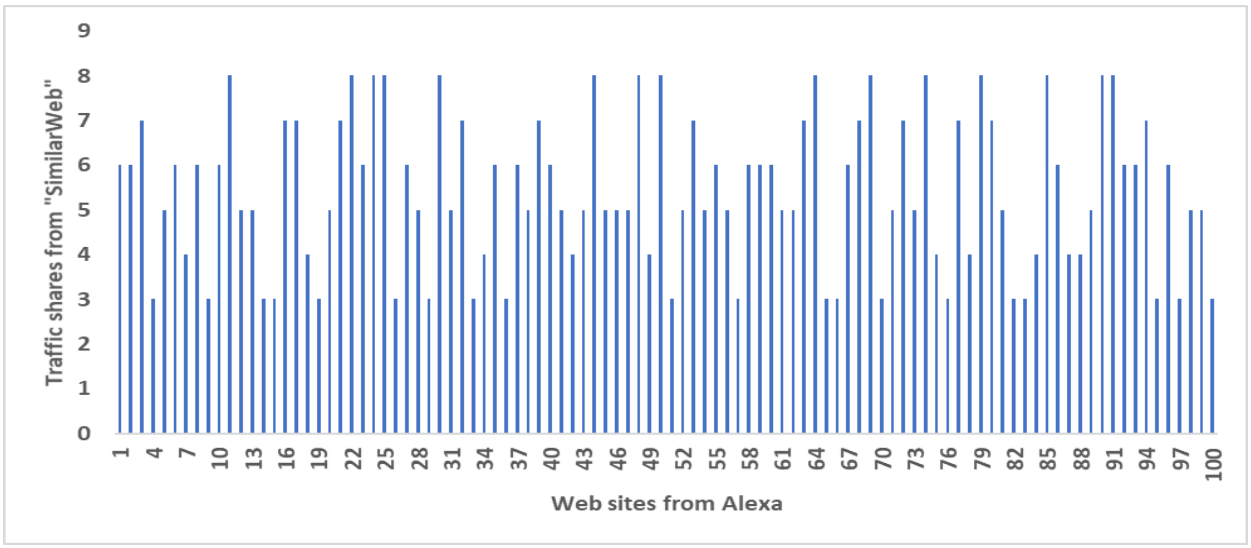
Figure 2. 3 Dataset produced from "Online Books Review" website

We also evaluate our work on the 500 popular websites ranked by "Alexa." We take the top 656 websites and remove 156 websites from the list. There are two reasons behind that. The first one is some of the websites do not meet the criteria defined in the case study section (We delete 97 websites from the list for this reason). As an example, "Google.ca" is very different from the other websites. We consider websites with more user interactions. The second reason behind that is, we need the number of views of any specific web pages to train our model. We use

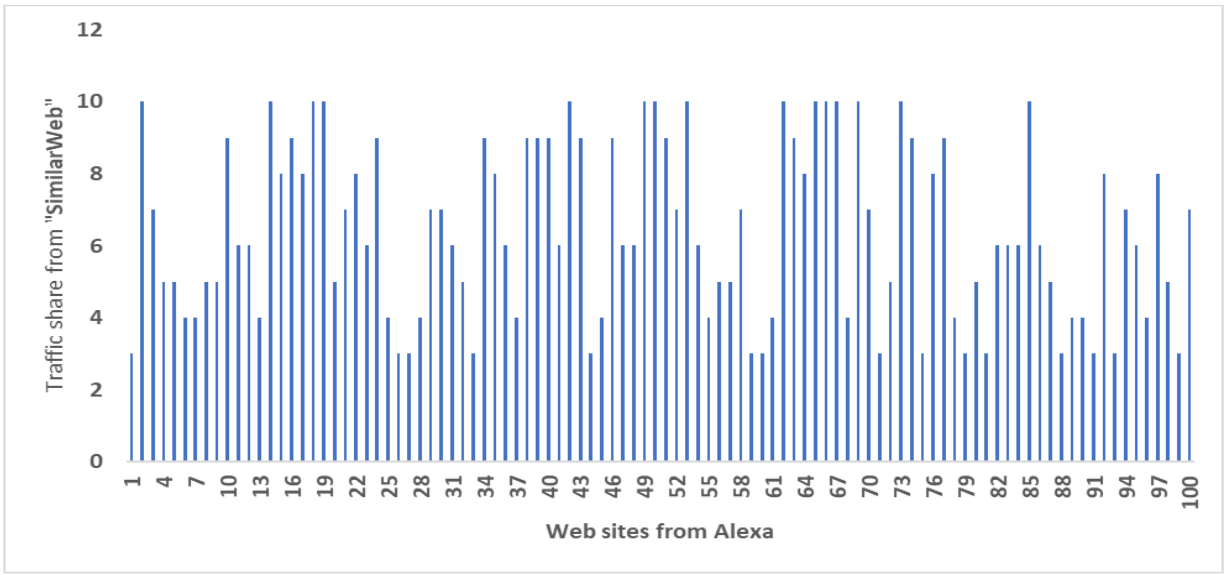
"SimilarWeb" to collect these "page views" as we don't have server access to the websites. Therefore, we select web pages for which "SimilarWeb" can generate these results. For 59 websites from the top 656 websites of Alexa, "SimilarWeb" fails to produce results; therefore, we discard them from the list. So after cleaning the websites dataset, we include 5 datasets; naming them "Dataset 4" to "Dataset 8", where Dataset 4 to Dataset 7 are used for training and Dataset 8 is used for testing. We have 489 web pages in Dataset 4, 540 in Dataset 5, 639 on Dataset 6, 659 on Dataset 7, and 611 in Dataset 8. Therefore, we have a total of **2,938** web pages in the dataset where **2327** web pages are used as training for our model and **611** web pages are used as testing. In figure 2.4 we represent for each site the number of pages we consider in our system.



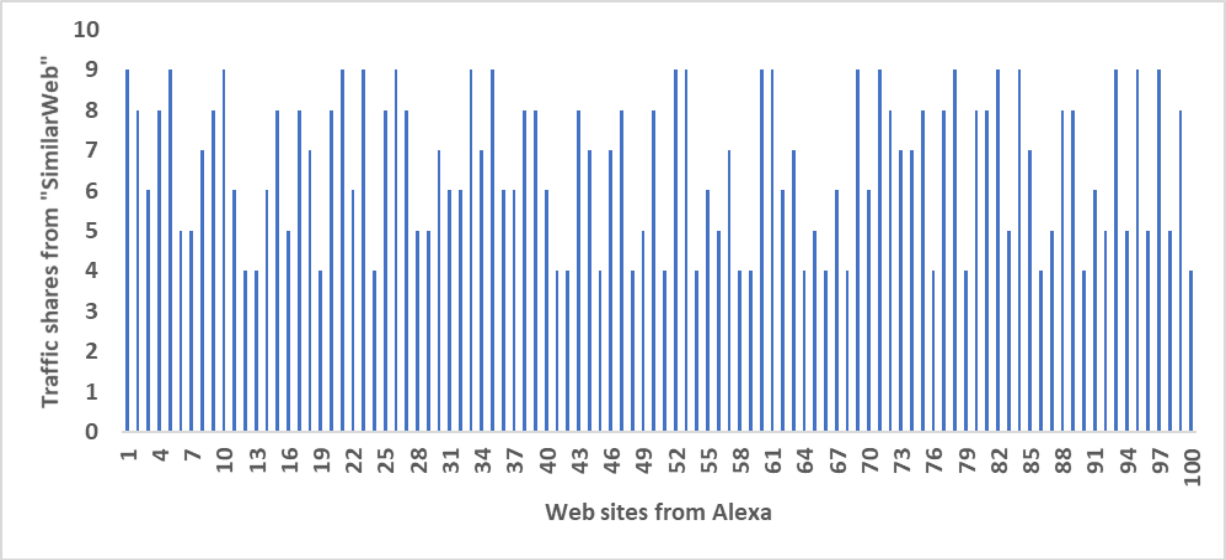
Dataset 4



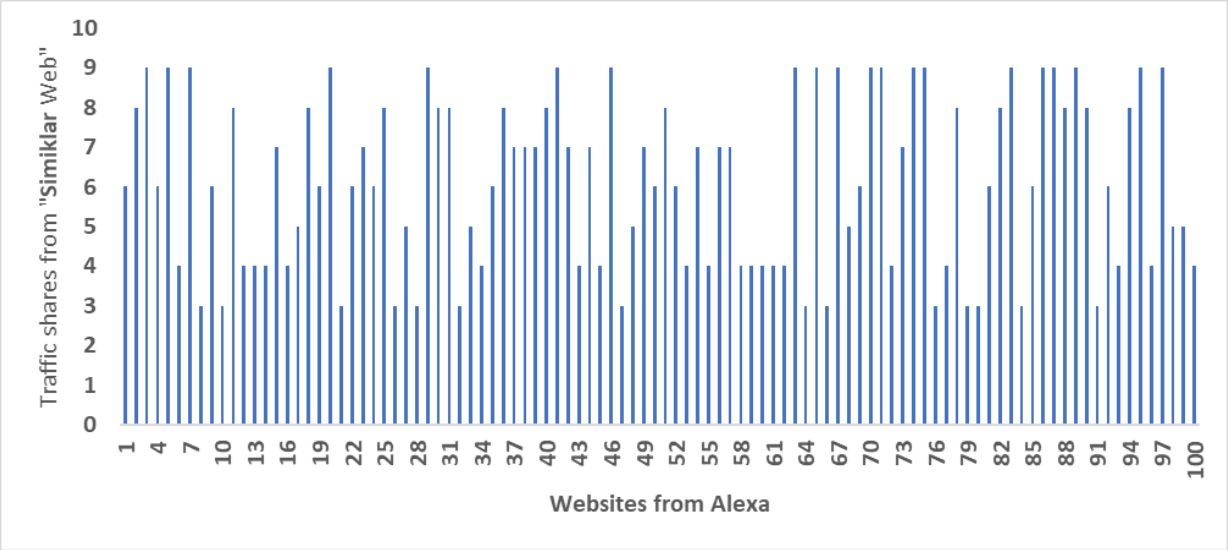
Dataset 5



Dataset 6



Dataset 7



Dataset 8

Figure 2. 4 Datasets produced from Alexa top 500 web sites

2.5.2 Experimental Results

Figure 2.5 shows the page views from "Google Analytics" versus the Importance Value produced by our system (Note that page views from "Google Analytics" are represented as $\log_2$ values.). The data shows potential clustering of the pages, "Terms of Services" had no Importance, while, as expected, "Home" page was mostly visited, and hence the most

important. It is observable that different types of Posts, Articles and Categories pages possessed higher Importance Values (I.V.). The values of I.V. correlate significantly with the page views from "Google Analytics" ($r=0.79$; $p<10^{-3}$). Applying a ranking of the page's procedure for both the page views from "Google Analytics" and Importance Value demonstrates a significant rank correlation $\rho=0.91$ ($p<2.2*10^{-16}$). **This provides important evidence that the Importance Value is performing similar to other analytic tools.**

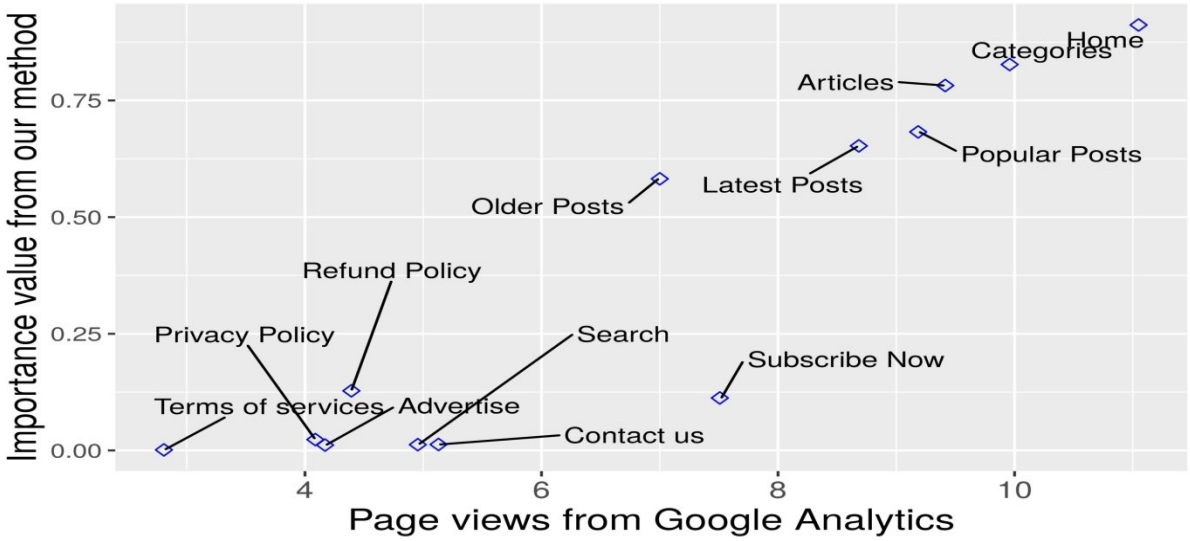


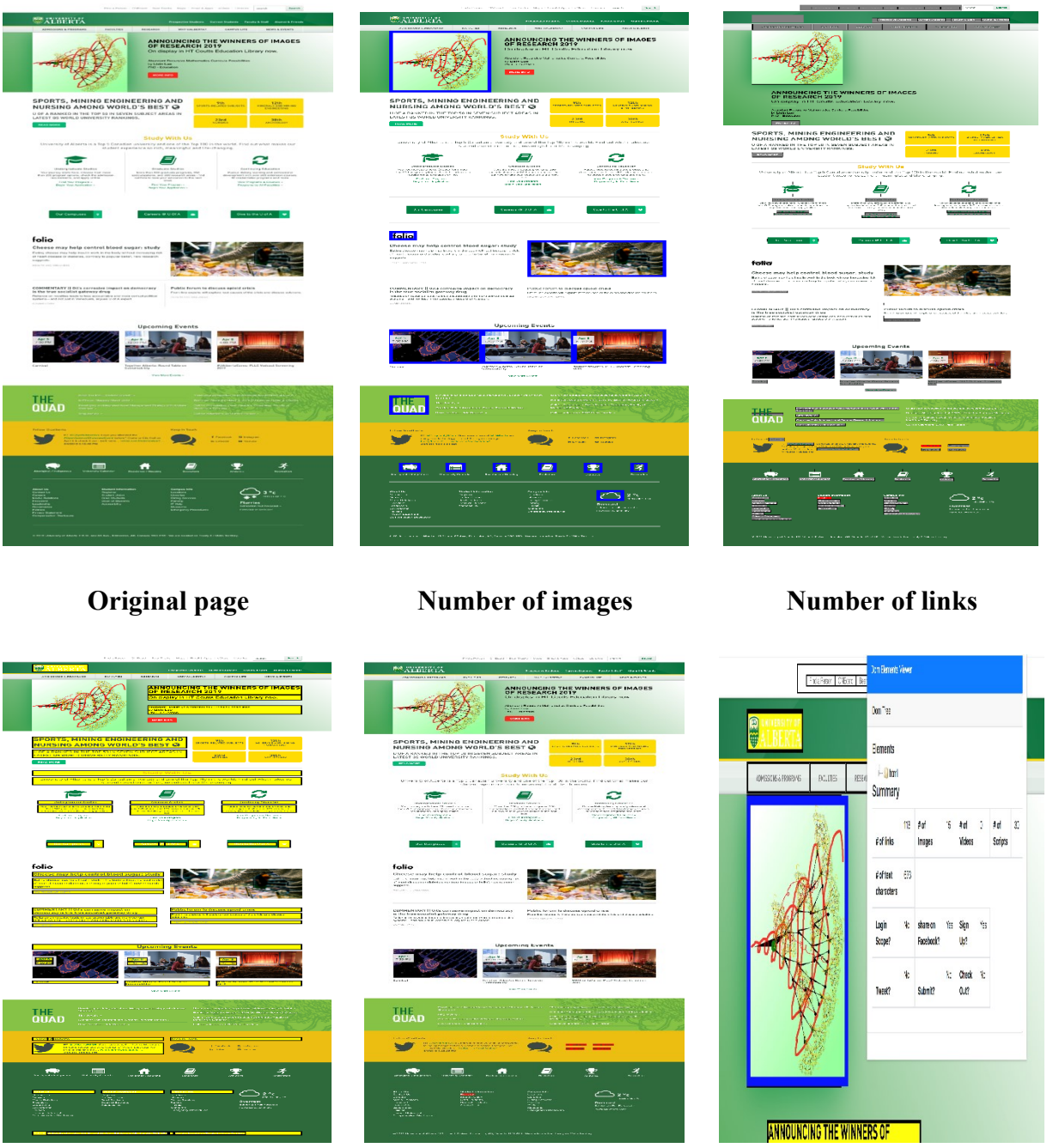
Figure 2. 5 Page views from "Google Analytics" versus the Importance Value produced by our system

2.5.3 Case Study

In this section, our proposed work focuses on three case studies. We also use another different website (University of Alberta) for this purpose rather than "Online Book Review" website. The reason for selecting this website is because of the vast amount of work that can be carried out here which gives us sufficient data to analyze.

Case Study-I: In Case-I, we show how our proposed system can extract the website contents. In figure 2.6(a) we represent the screenshot of the home pages of the Alberta website. In Figures 2.6(b) to 2.6(f) we represent the results. We represent images in 2.6(b), links in 2.6(c), and texts in 2.6(d). In figure 2.6(e) we show the integrations. From the figure, we see that this page has a

scope of registrations, sharable on “Facebook”, “LinkedIn”, and “YouTube”. Finally, in figure 2.6(f) we represent the total results in the extension window.

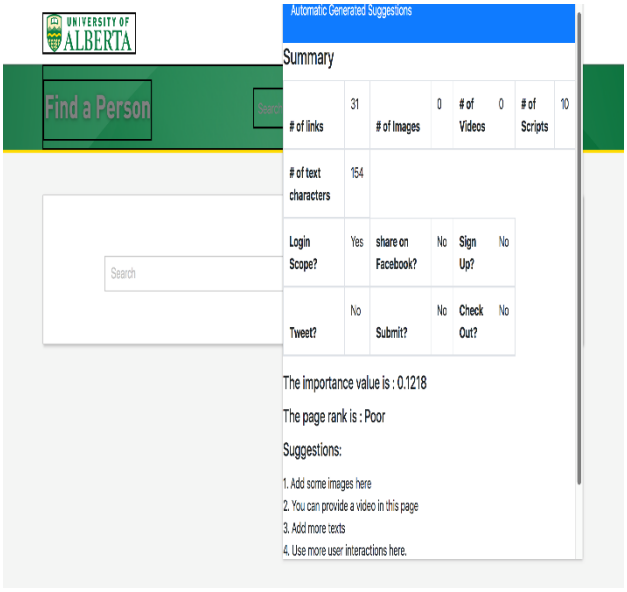


Original page **Number of images** **Number of links**

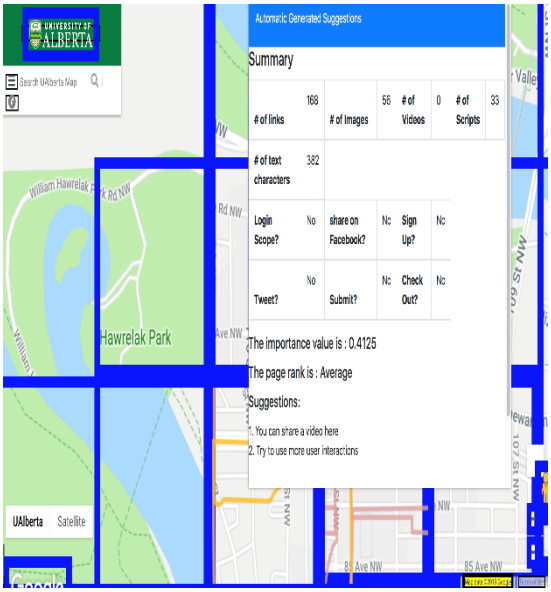
Number of texts **Interacted with users** **With results**

Figure 2. 6 Output of the extension for the Homepage of the website “University of Alberta”

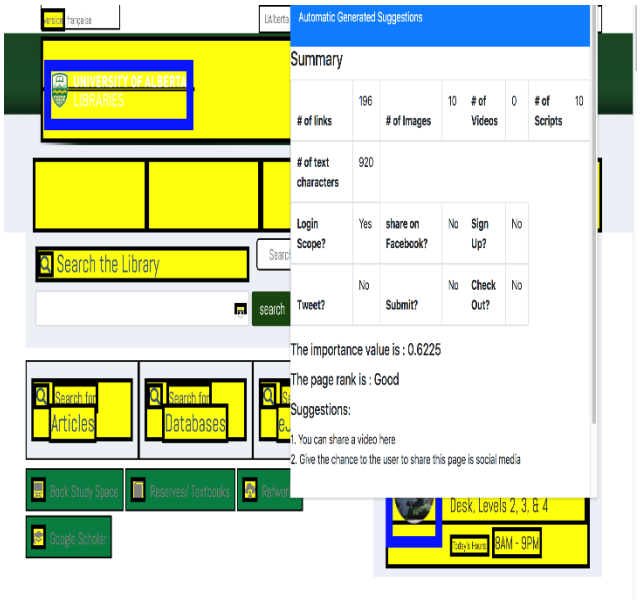
Case Study-II: In this case, we represent four different pages for which our system generates the four different scores; Poor, Average, Good and Best according to their value and importance.



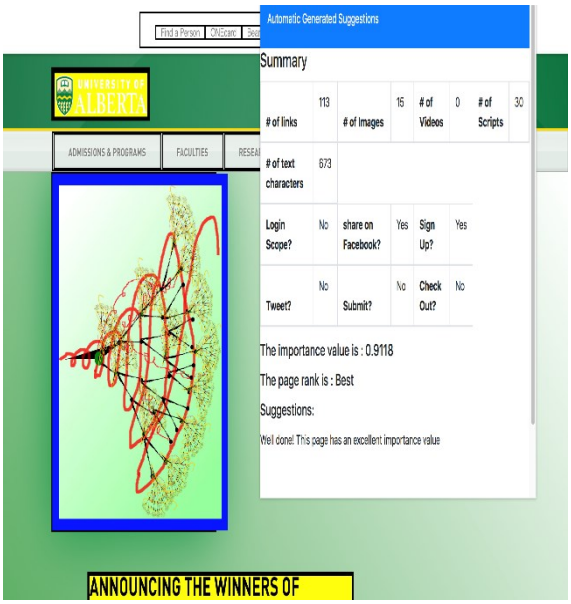
(a) Page with "Poor" ranking



(b) Page with "Average" ranking



(c) Page with "Good" ranking

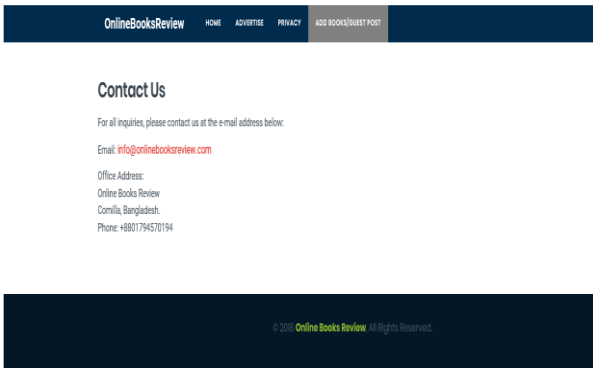


(d) Page with "Best" ranking

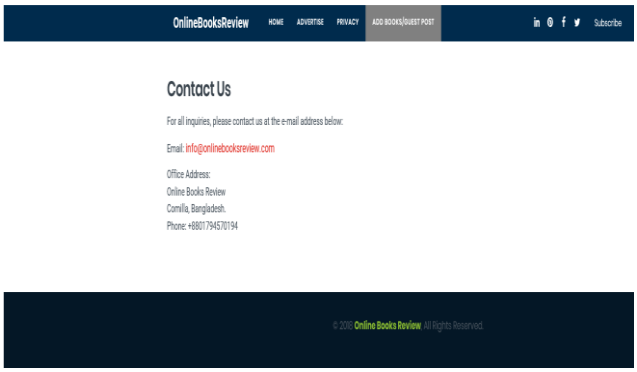
Figure 2. 7 Automatic suggestions provided by our proposed system

In figure 2.7(a) we see the page “Find a person” where our system produces its score as Poor and produces a very low importance value. We see there are only some texts and one interaction with the user. Our system produces suggestions for it to increase the score. In figure 2.7(b) the “average” scored web page “Map” is represented. The importance value produced by our system for this page is 0.4125. We see some suggestions here. In figure 2.7(c) the web page “Library” is represented with the results of our system. The importance value is 0.6225. So there are fewer suggestions for that. The Best rank given by our proposed system to the home page of the website is represented in figure 2.7(d). We see that the importance value is 0.9118 and there are no suggestions here. Our system can generate this suggestion automatically. These suggestions are reviewed manually and we find them very effective.

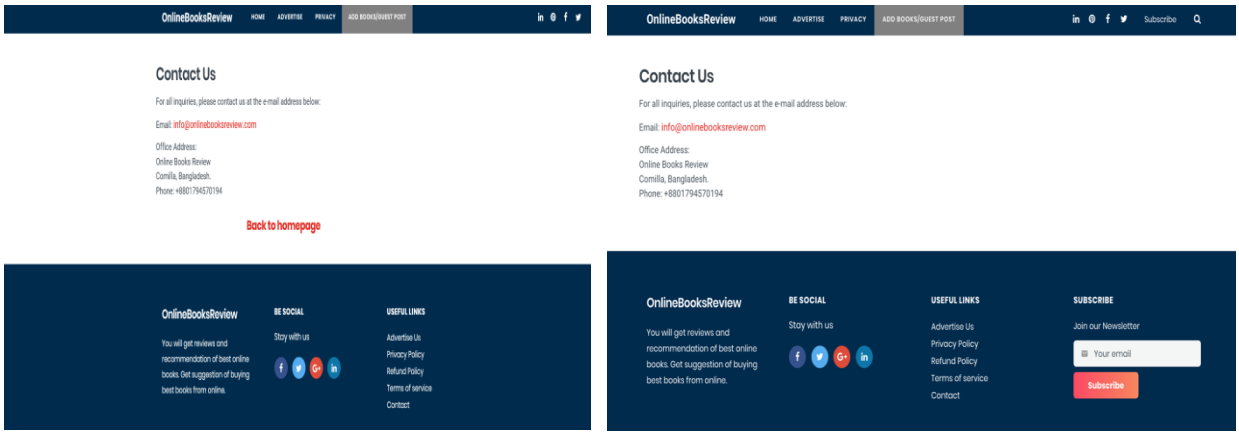
Case Study-III: A case study is also conducted on Online Book Review website. “Contact us” page(https://www.onlinebooksreview.com/contact) is chosen for this case study. Four web pages’ version are made. The webpages are then updated in these four versions. Figure 2.8 denotes the four versions.



(a) Version-1



(d) Version-2



© Version-3 (d) version-4

Figure 2. 8 Different versions of "Contact us" web page of "Online Book Review" website

The case study result is shown in Table 2.12. We can view the four versions of the features. Also, when we made the "contact us" page more interactive, there is an increase in the page views. So, this case study shows the effectiveness of our work.

Table 2. 12 Page views of "Contact Us" web page of "Online Book Review" website according to different versions

Features	Version number			
	1	2	3	4
Header with basic information only	yes	Yes	Yes	Yes
Header with a sharable link in social media	No	Yes	Yes	Yes
Header with subscribe option	No	Yes	No	Yes
Body with basic information only	Yes	Yes	Yes	Yes
Body with a back link to home page	No	No	Yes	No
Footer with more links only	No	No	Yes	Yes
Footer with a subscribe option with a mail address	No	No	No	Yes

Number of page views (according to "Google Analytics")	5	9	22	25

2.5.4 Validation of Results

For validation of our work, we use two types of validity; internal and external. In internal validity, we use the confusion matrix to represent the results, and for external validity, we use the correlation matrices; Pearson and Spearman.

2.5.4.1 Internal Validity

For internal validity, we represent our results in a confusion matrix. To find the internal validity we checked through all **2,938** web pages **manually** for their features. We extract the source code of all the 2,938 web pages and then check manually all the features and compare them with the automated generated results. We discovered how our system can find out the images, texts, videos, links, and user interactions efficiently. There are two basic measures used in evaluating the performance of these strategies. They are Precision and Recall. The recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage. On the other hand, Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. We also made use of four other parameters for more accurate analysis and to determine Accuracy, Precision and Recall for the extracted content. The parameters are:

1. True Positive (TP): The number of pages in which our system discovers where login scope truly exists.
2. True Negative (TN): The number of pages in which our system does not find the login scope where login scope truly exists.
3. False Positive (FP): The number of pages in which our system finds the login scope where login scope does not exist.

4. False Negative (FN): The number of pages in which our system does not find the login scope where login scope does not exist.

Here we give the example on the basis of finding the login scopes in a web page. We follow the same parameters for finding the images, videos, links etc. Then measure the Accuracy, Precision and Recall based on this result. We represent the results for both cases in table 2.13 to table 2.17. In the evaluation process at first, we go through each page manually at their source code and see how many images, texts, links, etc. are there. Then we compare these results with our automated generated system. In Dataset 1 to Dataset 3, we use the same web pages. So in each table, we represent the results of the 3 Datasets together. There are 100 pages in the 3 Datasets, and their design is not changed in the 3 months. So we represent the results together. For Dataset 4 to Dataset 8 we also represent the results in a confusion matrix. From the tables, we observe that our system can successfully extract the web pages' contents.

Table 2. 13 Evaluation for images of the web pages

		TP	FP	TN	FN	Accuracy	Precision	Recall
Online Book review	Dataset 1- Dataset 3	248	6	4	3	96.55%	0.9763	0.9841
Websites from Alexa	Dataset 4	1613	21	32	1	0.9682	0.9805	0.9871
	Dataset 5	2808	23	18	1	0.9933	0.9936	0.9918
	Dataset 6	1342	11	16	2	0.9868	0.9882	0.9918
	Dataset 7	3493	21	29	4	0.9906	0.9917	0.9940
	Dataset 8	1833	12	31	0	0.9834	0.9833	0.9934

Table 2. 14 Evaluation of Videos of the web pages

		TP	FP	TN	FN	Accuracy	Precision	Recall
Online Book review	Dataset 1- Dataset 3	8	0	1	96	99.04%	0.8889	1
Websites from Alexa	Dataset 4	13	0	0	481	100%	1	1
	Dataset 5	16	0	1	530	99.81%	0.9411	1
	Dataset 6	18	0	0	633	100%	1	1
	Dataset 7	16	0	0	648	100%	1	1
	Dataset 8	12	0	1	589	99.04%	0.8889	1

Table 2. 15 Evaluation of Links of the web pages

		TP	FP	TN	FN	Accuracy	Precision	Recall

Online Book review	Dataset 1– Dataset 3	422	11	5	9	96.42%	0.9882	0.9745
Websites from Alexa	Dataset 4	3182	54	24	44	97.63%	0.9925	0.9833
	Dataset 5	3672	402	27	49	89.66%	0.9927	0.9013
	Dataset 6	4473	490	32	56	89.66%	0.9928	0.9012
	Dataset 7	3823	417	34	62	89.59%	0.9911	0.9016
	Dataset 8	3178	348	22	55	96.42%	0.9882	0.9745

Table 2. 16 Evaluation of words of the web pages

		TP	FP	TN	FN	Accuracy	Precision	Recall

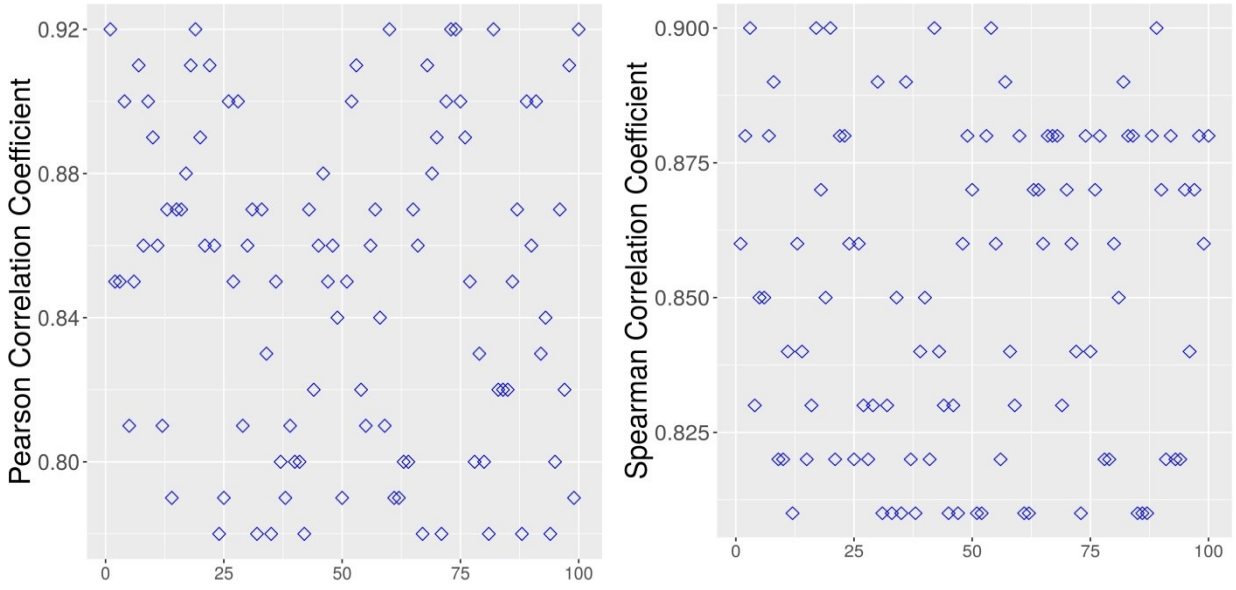
Online Book review	Dataset 1- Dataset 3	8665	256	11	9	97.01%	0.9987	0.9713
Websites from Alexa	Dataset 4	42380	1220	42	51	97.11%	0.9990	0.9720
	Dataset 5	46990	1311	49	62	97.19%	0.9989	0.9728
	Dataset 6	55602	1492	58	58	97.29%	0.9989	0.9738
	Dataset 7	57419	1598	61	65	97.19%	0.9989	0.9729
	Dataset 8	54235	1482	51	56	97.01%	0.9987	0.9713

Table 2. 17 Evaluation of User interactions on the web pages

		TP	FP	TN	FN	Accuracy	Precision	Recall
Online Book review	Dataset 1- Dataset 3	398	5	2	0	98.27%	0.995	0.9875
Websites from Alexa	Dataset 4	1908	28	482	11	79%	0.7983	0.9855
	Dataset 5	2112	45	392	17	82.96%	0.8434	0.9791
	Dataset 6	2710	62	401	29	85.54%	0.8711	0.9776
	Dataset 7	2882	82	445	21	84.63%	0.8662	0.9723
	Dataset 8	2502	39	312	31	87.82%	0.8891	0.9846

2.5.4.2 External Validity

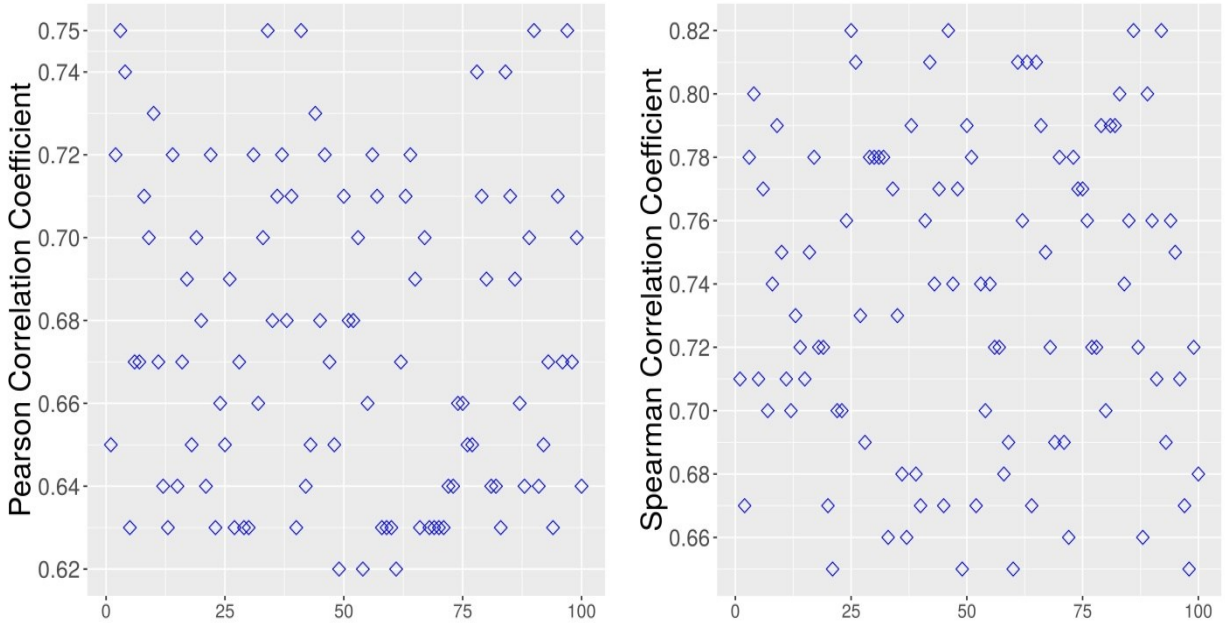
For representing the external validity, we use the two types of the correlation coefficient: Pearson and Spearman. We find the correlation among pairs of variables; the first one is the importance score produced automatically by our proposed system and the second variable is the "page views" results collected from the "Google Analytics" and "SimilarWeb." We use the Dataset 3 and Dataset 8 results to represent the correlation as they are used in our proposed system as testing. Pearson Correlation Coefficient is represented by r, which originally stood for regression. It is a parametric statistical measure of the strength of a linear relationship between two continuous variables. A relationship is linear when a change in one variable is associated with a proportional change in the other variable. Pearson Correlation Coefficient examines the variables concerning their deviations from the mean. On the other hand, Spearman's rank correlation coefficient is a nonparametric rank statistic proposed as a measure of the strength of the association between two variables. It is a measure of a monotone association that is used when the distribution of data makes Pearson's correlation coefficient undesirable or misleading. It assesses how well an arbitrary monotonic function can describe the relationship between two variables, without making any assumptions about the frequency distribution of the variables. Unlike the Pearson correlation coefficient, the Spearman correlation coefficient does not require the assumption that the relationship between the variables is linear, nor does it require the variables to be measured on interval scales; it can be used for variables measured at the ordinal level. However, the sign of the correlation tells something about the behavior of the two variables; the absolute value of the correlation indicates how strong the relationship is between these variables. A correlation of 1.0 is a perfect positive correlation, meaning that the two variables move upward or downward together. A correlation of -1.0 is a perfect negative correlation, meaning that the two variables move in opposite directions. The closer the correlation is to 1.0 or -1.0, the stronger the relationship between the two variables. The sign only determines the direction, positive or negative, and it does not influence the strength of the correlation. When there is no linear correlation between the variables, the value of the correlation coefficient would be 0.



a) The Pearson Correlation

b) The Spearman Correlation

Figure 2. 9 The Correlation among pairs of variables in our proposed system's score and "Google Analytics" page view



a) The Pearson Correlation

b) The Spearman Correlation

Figure 2. 10 The Correlation among pairs of variables in our proposed system's score and "Similarweb" page view

We use both cases (The website of “Online Book Review” and the Top websites from “Alexa”) for finding external validity. In the case of “Online Book Review” website, we consider the “page view data from the “Google Analytics” for four weeks of December 2021. Suppose for the first web page, we first produce the importance value automatically. According to our system, the importance value will not change in four weeks as the website design is not changed. So we select four different values of page views collected from “Google Analytics” and four unchanged values automatically generated from our proposed system. After that, we use this data to find the Spearman and Pearson Correlation coefficient. In this way, we go through the remaining 99 web pages and generate the correlation values. Figure 2.9(a) represents the Pearson Correlation coefficient, and Figure 2.9(b) represents the Spearman Correlation coefficients for the “Online Book Review” websites. Estimation of Pearson correlation coefficients denoted strong correspondence between variables. The values varied between 0.78 and 0.92, with a mean of 0.85. On the other hand, estimation done applying Spearman correlation evidenced a strong correlation between variables. This statistical parameter varied from 0.81 to 0.9. On average its value was 0.85. So our system can find the importance value of “Online Book Review” website successfully. In the case of top websites from “Alexa” we use Dataset 8 where we also find 100 websites. For each website, we generate the importance value of the web pages automatically and then use the results of “SimilarWeb” to compare. In this way, we generate for 100 websites and represent the Pearson Correlation in Figure 2.10(a) and Spearman in Figure 2.10(b). In the case of Person Correlation, the correspondence between variables is calculated between 0.62 and 0.75. Association between the variables, expressed by average correlation was 0.67. Spearman coefficients values are estimated as not less than 0.65. The average correspondence between analyzed variables is 0.74. The maximal correlation identified is 0.82. So we can conclude that our system can generate similar results to that of “Google Analytics” and “SimilarWeb.”

2.5.5 State-of-the-art

The research carried out by Thomas et al. [2019] is similar to our research. An experimental study was carried out to assess social commerce's impact on website features in their research. Four versions of a website were created and they use for testing purposes. The feature of the fourth version is richer than the other three versions tested. Below are a few comparative studies between

our work and that of Thomas et al. [2019].

- The website's high-level issue was worked on by Thomas et. al. [2019]. This issue varies for different users. Some of the issues considered include:
  i. Perceived satisfaction
  ii. Perceived usefulness
  iii. Trust
  iv. Operation checks items.

  These issues are varied on the website to view users' responses to increasing or reduction. The responses are recorded with a "Yes", "No" or "Unsure". The numbers of clicks, page views per user and time spent were also recorded. However, our model focuses more on the website's low-level features to observe the responses of users to changes in features. Therefore, our research encompasses almost all website features.
- In Thomas et. Al. [2019] experiments, **4 website** version was used namely "zero", "low", "medium" and "high" versions. The richness of each feature was in ascending order from zero level to high version. In our work, we chose the selected Alexa top **500** websites from the top 656 websites while the "Online Book Review" website was also considered since we have access to its server.
- They receive **212** participants' feedback with a significant number of them **164** were used and some were discarded. Likewise, we also keep track of web page users' responses through data generated from "SimilarWeb" and "Alexa". For instance, "SimilarWeb" was able to track about 1 Million Amazon website users. This gives us a robust amount of real-time participants.

2.6 SUMMARY & CONCLUSION

Web applications have infiltrated almost every aspect of our daily life. Research shows that 93% of online shopping starts from websites search. Therefore, to capture the online marketplace places the advertisement provider needs to know the right place to set up their ad so that most of the website users can see their ad. There are lots of web applications available capable of fulfilling this purpose but most of them use web users' private data. So, when users close their browser

cookies information the web applications won't be able to get accurate results. The main highlight and fascinating aspect of our work are that it works on the website structure to predict the importance of the web page. It consists of two very helpful features for both web developers and advertisement providers.

- In the case of an Advertisement provider, our proposed system can show the importance score alongside the web pages' rank so that they can take a quick decision to include their advertisement in real-time. No user private data is needed.
- In the case of web developers that sometimes publish their trial version and later use feedback gotten from the users to update their web application. Our system can give them real-time suggestions with the importance score so they can design a better website in the development period.

For solving the problem, we extract the features from web pages in real-time and use CatBoost Machine learning to create the rank. We do not only use the web pages' contents (such as the number of images, number of videos, number of links, number of texts, etc.) but we also use the web page accessibility and connectivity with other web pages. To validate our work, we use two types of datasets; one is collected from the server of the "Online Book Review" website and another we prepare from the most popular 500 websites from Alexa. We represent our effectiveness in the format of case studies, confusion matrix and correlation coefficient. In all formats, our good results prove the effectiveness of our proposed system.

Chapter 3: Real-time Browsing Assistant on Web

ABSTRACT

Understanding user requirements based on their interactions with a website is becoming increasingly important. Hence, in this paper, a novel real-time navigation-support system is discussed. This system builds a personalized browsing assistant based on the current user request submitted to a web server. The process involves developing a behavior model using a Discrete Time Markov Chain (DTMCs) inference process. This is then used to monitor user activities, and thereafter suggest “where to go next”. Finally, it updates the model in real time using a Markovian Decision Process (MDP). To evaluate the system, we provide a user study, case studies and conduct experiments on two datasets to verify the effectiveness of our proposed system.

3.1 INTRODUCTION

According to research, 93% of purchase decisions start with using a search engine. Sometimes we fail to get the best services from websites because not all websites are developed according to users' demands. It is not possible to design a website in accordance with the users' requirements as there are a large number of users and their demands evolve over time. Hence, website users have a hard time searching for web pages that they need on a website. To solve this issue, we have developed an automated suggestion system for website users to allow them to optimally traverse a website. Our system will suggest to users their next navigation steps (hyperlink following) according to previous activity on the website from previous users.

We develop an interactive system that can interact in real-time with users. Our system will enable us to incrementally generate user-behavior models based on user-intensive web application browsing. Specifically, it takes user navigation patterns as input data and generates an inference model of the website using a Discrete Time Markov Chain (DTMC) process that is continuously updated using Reinforcement learning (RL). In the inference model, the nodes are the unique links of the website, and the edges are the transition probabilities of moving between links. By analyzing

the transition probabilities, we predict the users' appropriate navigation steps. This is realized by building a real-time system that can generate suggestions by taking the user's requested link as input and providing appropriate suggestions.

Our paper makes the following major contributions:

- We build a real-time suggestion generation system for websites that can be used as an add-on to web browsers. If users find their expected link on the additional suggestion bar they can simply click and be redirected. This improves the user experience since more accurate navigation data is presented.
- As a user searching for content online might take a lot of time, this has been eased in the proposed system by providing screenshots of the suggested web links/web pages. This gives the user a glimpse view of the website before opening and exploring it.
- A user study with two test cases is conducted in this study, one with our extension and a different one without the extension. This is used in a practical setting where users provide feedback on the usability of the two. This is then tabulated to prove the proposed system's effectiveness.
- The paper evaluates case studies, and the "University of Alberta" website, to demonstrate the effectiveness of our tool in improving the user experience.
- Finally, the results are evaluated and a cross-validation process is conducted to ensure the system produces the desired results.

In section 3.2, we discuss briefly previous research that is related to our work. Then in section 3.3, we discuss the methodology of our proposed approach. After that, in section 3.4, we evaluate our system via case studies, user studies, prediction results and cross-validation. Finally, we summarize the paper and arrive at our conclusions in section 5.

3.2 RELATED WORK

Much work has been done on Web Usage Mining (Adeniyi et. al. [2016], Wang et. al. [2016]). In general, three major orientations can be found in this research area: analyzing user behaviors, clustering the users of a website and web link prediction and recommendation. We represent the most recent works here.

In the first category, most of the research has been done on how users react to different links of a website. Nagy et al. [2009] provides a clustering approach to make group of similar web pages by distributions of spent times. The distribution of this spent time is different at dissimilar types of page such as registration form, index pages, news, description of products etc. Users spend more or less time to read several pages; they apply distribution of spent time to find the correspondence of documents of the site. They test their approach on log files generated by a commercial website. Schur et al. [2013] present a fully automated tool that mines explicit behavior models of enterprise web applications for system testing and maintenance. They use a conference web sites log file as their test bed, they claim that their automated system can produce models from the web site that can be directly used for effective model-based regression testing. The main objective Arpakis et al. [2014] is to understand the potential impact of response latency on users search behavior. They describe the dominant factor in web search and demonstrate the relative importance of each factor using real life data traces. They conduct a small scale, controlled user study which reveals the difference in the way users perceive the latency. They also conduct a large-scale analysis using a query log obtained from Yahoo search. Guan et al. [2014] analyze the behavior of the user of the micro-blogging website named "Sina Weibo". They select 21 social hot events that are widely discussed on "Sina Weibo" in 2011. They empirically analyze the users posting and reposting characteristics. They find that the reposting (making comments under a post) rate is three times higher than the posting rate in the blogging site and males are more actively involved than females in the time of social events like "Rock and Ring musical events", "Soccer games tournament" etc. Ghezzi et al. [2014] present an approach that automates the acquisition of user-interaction requirements in an incremental and reflective way. Their solution builds upon inferring a set of probabilistic Markov models of the user's navigational behaviors. They extract the navigation history from the log file of a (small and imitation) web application www.findyourhouse.com. They annotate and analyze the inferred models to verify the quantitative properties employing probabilistic model checking.

In the category of clustering users, researchers normally cluster websites based on user actions. This can also be achieved through the clustering of the clickstream data. Banerjee et al. [2009] propose an algorithm for clustering the website users based on a function of the longest common subsequence of their clickstream that takes into account both the trajectory taken through a website and the time spent at each page. They use the weblogs of www.sulekha.com to illustrate

their technique and present the results. Wan et al. [2010] transfer the clustering task into a chaotic optimization problem by proposing a CAS-based clustering algorithm. They claim that they propose this type of algorithm first time in web user clustering. They compare their proposed algorithm with the most classic k-means clustering algorithm in terms of average intra-cluster distance and average inter-cluster distance. Gang et al. [2016] bring a new era to this type of research by making that cluster visible. They identify the clusters of similar users by partitioning a similarity graph where nodes are the users of the web system and edges are the weighted clickstream similarity. The partitioning process leverages iterative feature pruning to capture the natural hierarchy within the user cluster and produce features for visualizing and understanding captured user behaviors. For evaluating their system, they present two case studies on two large-scale clickstream traces from social networks. They can identify the dormant users, and hostile chatters (the user of the society who tends to block other people during chatting) in their system.

In the third area of web link prediction and recommendation, Shahriary et al. [2015] propose a ranking algorithm for detecting the community in signed graphs. They test their algorithm on three large-scale datasets; Epinions, Slashdot and Wikipedia. Liu et al. [2007] propose an approach to classifying user navigation patterns and predicting the user's future requests. Their approach is based on the combined mining of web server logs and the content of retrieved web pages. They capture the textual content of the web page and then they use the character of N-grams to represent the content of the web pages. Then they combine it with web server log files to derive user navigation profiles. Javari et al. [2014] propose a new method for sign prediction in networks with positive and negative links. Their algorithm is based on first clustering the network into several clusters and then applying a collaborative filtering algorithm. Then they use the similarity between the clusters based on the links between them. Tan et al. [2018] focus on App usage prediction based on link prediction in bipartite networks. Their main task is to predict whether a user will use an App or not based on the historical NFP (Network footprint) data. They construct User-App bipartite network and transform the App-usage prediction into a link prediction problem in the complex network which can focus on extracting missing information. For testing, they collect 4-days NFP data from ISP's Operational network. Gurini [2015] et al. exploit sentiment analysis for identifying latent communities and their subsequent use in recommending similar users. They provide these recommendations to the target users for better networking on social media. Adeniyi [2016] et al. presents a study of automatic web usage data mining and recommendation system

based on current user behavior through their clickstream data. They use a K-Nearest-Neighbor (KNN) classification method for training the model and matching it to a particular user group for a particular period. To achieve this, they extract the RSS address file, clean the file, format the file and finally group the file according to their session. Wang [2008] et al. developed an online navigation aid using collaborative recommendations based on graph theory. They utilize the past data of the user for that and also use the server log file as input.

Up until recently, web link prediction and recommendation are done using website log files as the input. This means that if the website is updated then the system does not work. Another challenge is the accumulation of log files that occur very fast. This leads to the algorithm being inaccurate due to the huge work needed for processing a log file. This is mainly contributed by the factor that as the log files increase in size the algorithm will continue using the data of the oldest to the newest while older versions are not necessarily required in making suggestions. This has been solved in the new system where real-time suggestions are done based on behavioral models that are updated in real-time. Besides this, the recommendations of previous studies are conducted offline and can therefore only be accessed by the website developer. But, as our system is interacted in real-time with the users it can give suggestions to the users too. The model generation procedure of Ghezzi et al. [2014] and Emam et al. [2018] is related to the proposed system framework. However, their proposed system utilizes old log files, which means that it does not interact with the website users fully and not in real-time.

3.3 OUR PROPOSED SYSTEM

This section of the paper covers the proposed system in detail. It has been discussed above that the system works in real-time. Essentially a website collects users' requests as input. The system utilizes an extension to make an accurate prediction of where the user may want to go next. The extension can be installed as an add-on in the web browser. There are four basic steps in our proposed system:

- Collecting user requests as input and pre-processing them.
- Generating and continuously updating behaviour models of the user’s interactions with the website.

- Updating the models solely based on the importance of the requested URLs. (These models can be extended if other types of information are transmitted.)
- Resolving ambiguities in the models to generate accurate suggestions and next steps for the user.

3.3.1 Collecting user requests and pre-processing them

Tracking the user request is done in real-time. This is achieved by using the function console.log () at the JavaScript level. We observe several basic items in the user request – the name of the webserver and the encrypted IP address of the user, the timestamps, requested URL, name of the browser, name of the operating system and the user device information.

The timestamp is used to identify the session of the user. A session in this case is the number of interactions that occur on a website within a given time frame by an individual user. This means that in a single session a user can take multiple actions on a site. A single user can as well open different multiple sessions which range from hours to months. Campaign change and time-based, proposed by google analytics, are the two methods that are used to determine an end of a session. As soon as a user opens a site, back counting starts with a time constraint. For all the sessions started by the user, a time constraint is updated. The session remains active until the allowable time with inactive is expired. All activities by the user after this session are considered as a new session.

The system classifies users according to their browser and operating system utilized; different classes have a unique combination of browser and operating system. (if the proposed system of server support three browsers and three operating systems then the system will generate nine user classes and models.) A Discrete Time Markov Chain inference process is used to generate the user behavior model after the session and user classes have been identified. Algorithm 1 represents the overall procedure of section 3.1.

Algorithm 1. Preprocessing the user request

Input: User request

Output: a number of DTMC

1. Repeat the following steps until no new request is found.

2. Extract the Timestamp(TS), Browser name(B), Operating system(OS), and Requested link (URL).

3. Identify the session of the user requests.

4. **if** the session is old **then**

5. Update the model (details in section 3.2)

6. **else**

7. find the user class according to the browser (B) and operating system(OS).

8. **if** the class already exists in the system **then**

9. update the model for that class

10. **else**

11. start generating a new model for a new class.

3.3.2 Generating and continuously updating behaviour models of the user's interactions with the website

This section covers the generation of the users' behavioural model. As indicated above this is based on the Discrete Time Markov Chain inference process where the system works in real-time by generating a user-specific behaviour model for each user class. The Discrete Time Markov Chain (DTMC) used in this section has the following advantages;

- Every transition is dependent on the current state, this means that the usage patterns are clearly illustrated. The probability distribution of the next page is therefore reliant on the page the user is currently interacting with.
- The system utilizes a discrete-time interval and as a result, changes do not occur aimlessly.

DTMC normally has four initial parameters. The parameters for the proposed system can be defined as follows;

- S is the set of states. In our proposed system, URLs are considered as the state of the system.
- $P: S \times S \rightarrow [0,1]$, the probabilistic matrix indicating the probability of the occurrence of a transition between two connected states. Suppose, "Home" and "Contacts" are the URLs and from 100 requests by the website users we find 25 requests of the URL "Contacts" from "Home". ("XXX" refers to examples from the University of Alberta case study. This site can be viewed at: www.ualberta.ca). Then the probability between the links "Home" to "Contacts" is 25/100 or 0.25. Initially, this matrix is 2X2 where two states are "Start" and "End". If we find three URL requests, then the matrix will be 5X5.
- L is the function that is used for levelling the state of the DTMC. It starts with Label, L= {Start, End}, so if a new URL comes in at the requested time a new L is added. In our system, we level the states with their URLs.
- ρ is reward function that associates a non-negative number to each state. A reward is a numeric value that is annotated to a DTMC. The reward indicates the advantage in a quantitative value of visiting a page of the website or being in a particular state of the model. Initially, we set the reward value as zero.

Since our behavioral models are generated in real-time, the system can associate the reward values from the current state of the model incrementally. This is discussed in section 3.3 in detail. Below is an introduction to the inference model.

- The inference process initializes the model with a "Start" and an "end". It then considers the request. If the requested URL is from a new session, the system adds the new state to the model. The new state is labelled by the name of the new URL. In this case, the system considers the start state as its parent state.

- In cases where the requests belong to an old session, the system generates a state with the same labels, but it will consider the current state or the new state as its parent.
- A transition probability, P_{ij} is assigned to the transition between S_i and S_j, which is equal to the ratio between the number of transitions between the state, S_i and the state, S_j and the total number of transitions
- If the session expires during the inference procedure, the system generates a transition from the current state to the end state and updates the transition probability.
- The above steps will be repeated until no new request is sent by the user to the website during the current session.

3.3.3 Updating the models solely based on the importance of the requested URLs

The system relies on the DTMC for making decisions on the user's future activities on the website. This however develops a challenge for steady-state calculations in the DTMC. A steady-state means that we have reached a point where the distribution will no longer change. This means that the probability matrix is no longer effective since it will produce constant, or (highly) similar, result for the suggestions of future activity. This is where the Markovian decision process comes in handy. Reinforcement learning is integrated into the Markovian modelling approach adopted in this work— this approach is inspired by Emam et al. [25] for solving the above issue. We set the reward value to zero at first for the current state and then according to the action taken by the user, in real-time (the link of the web page clicked by the user), we update the reward value of the current state (the web page that the user currently views) immediately.

To illustrate this process, we will consider a buying and selling-based website. The goal of such a website would be to increase the adverts presented. If this is the case, the designer can typically assign reward values to states by considering the number of adverts on a page. This means for a typical page such as "Contact Us" with 5 adverts will equate to a reward value of 5 with the web link of the "Contact Us" page. For our proposed system, we track the content of web pages in the automatic reward calculation process (the details are given in algorithm 4)

The above issues are used to update the system in real-time; therefore, require a reward function. The purpose of the reward function is to specify the reward for every action that the users

of the website are performing. However, the main goal is to maximize the total reward values in the long-term view, this is achieved by reinforcement learning (RL). RL can learn what is required to maximize a numerical reward signal since it is located between supervised and unsupervised learning. The process normally involves trial and error since the learner is not aware whether the right procedure is achieving the goal of maximizing the reward signal. RL is therefore characterized by learning a problem. We involve the Markov Decision Process (MDP) where a user (who can be referred to as an agent) selects an action and the proposed system (which can be referred to as the environment) responds by presenting new states. The MDP, therefore, models the decision-making process. Essentially the MDP is a framework that provides a mathematical framework that is used to develop a model decision-making process where the outcomes are partly random and partly the decision of the user. (MDPs are similar to Markov chains with differences in additional actions which enable choice and rewards giving motivation.) This means that hypothetically if there is only one action for each state, there should be similar rewards. A Markov chain is therefore a derivative of the MDP due to the consistency in the process, in other words, the MDP can be said to be a discrete-time stochastic control process. In our proposed system, the MDP contains;

- A set of possible state ,S; in our case the URLs of the web page.
- A set of possible actions A. Here we consider the next web pages that are reachable from the current web page as actions.
- $P_a(S_t, S_{t+1}) = P_r(S_{t+1}, |S_t, a_t = a)$ is the probability that an action, a in state, s at time t that will lead to state, S_{t+1} at time t+1. In, if S is the URL of the web page "Home" at time t and $S_{t+1}$is the URL of the web page "Library" at time t+1 then a is the transition probability of going to page "Library" from page Home" at time t.
- $R_a(S, S_{t+1})$ is the immediate reward received after transitioning from state 'S' to state 'S_{t+1}' due to an action, a, at time t.
- γ is the discount factor that can be located from 0 to 1, which represents the difference in importance between future rewards and present rewards.

MDP, from the formulas above, can therefore be referred to as a set of states. As discussed this is the outcome that has been influenced by an action. Take for instance a drone navigating through a building, the state can be considered as a building or a house, or the x and y coordinates.

The MDP has a set of actions as well. A decision maker may choose any action 'a' that is available in state 'S' since at every other point in time the process is in some state 'S'. The process randomly moves into a new state 'S_t' at the next time-step. This provides the decision-maker with a corresponding reward 'r' (S, a, S_t). Based on this definition of a Markovian process, if a user on the web page "Home" at time 't' clicks on the link "Library" and reaches that page we can present it as;

$$P(s_{t+1}|s_t, s_{t-1}, s_{t-2}, \dots) = P(s_{t+1}|s_t) = T(s_t, a_t, s_{t+1}) \quad (1)$$

Where,

$P(s_{t+1})$ = P("Library")= The probability that a user is on the "Library" page.

$P(s_t)$ = P("Home")= The probability that a user is on the "Home" page

a_t= User clicks the web page "Library" when he is at the "Home" page

The above illustration depicts that the probability of the system moving into a new state is influenced by the chosen action of the user of our system. This process is known as the Markov property which means that the next state 's_{t+1}' depends on the current state 'S_t' and the decision maker's action "a_t". The previous state and action, therefore, influence the next state and therefore an immediate reward. The immediate reward is achieved by the agent observing the state $S_t \in$ S, choosing an action $a_t \in$ A at each discrete time, and therefore receiving an immediate reward r $\in$ R. This means that the state changes to S_{t+1} . This means a typical user receives an immediate feedback value after visiting a particular web page which means they reach a certain state. Since the progress of the RL algorithm is typically iterative, the agent normally learns during different iterations by observing the current environment state and executing an action. This is what guides the agent to the next state. In general, the above process can be summarized in the following formula;

- The value function, $V^{\pi}(s)$, specifies "how good" it is for the agent (users of the website) to be in a given state (URL of the website). We express the "How good" notation in terms of the future reward (importance of the webpage to the website). We expect the reward

signal will be maximized. Therefore, we can define the value of a state, S under a policy, π by using the equation:

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\} \qquad (2)$$

Where π is the stochastic policy between 0 and 1. This is a mapping from each state s and action a to the probability $\pi(s, a)$ (transition probability from a state S to S_t). E_{π} is the expected return earned by following policy π and discount factor γ, $0 \leq \gamma < 1$, which models the fact that future rewards are worth less than an immediate reward.

- We then find out the value of performing an action. When we are in the current state S and we have some possible future states, we can then find which actions are "How good" for the users of the website. This value of performing an action, a, in state S can be defined as:

$$Q^{\pi}(s, a) = E_{\pi}\{R_t | s_t = s, a_t = a\} = E_{\pi}\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\} \quad (3)$$

We can find the value of actions from a current state by this equation. While this process is iterative, in our proposed system there is a lack of known transition and reward models. It is therefore important that there is some sampling and exploration to learn the required model. In this case, we utilize Q-Learning, which is used to estimate the Q-value function in a model-free fashion. This can be represented as;

$$Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q_k(s_t, a) - Q_k(s_t, a_t) \right) \quad (4)$$

Where, α is the learning rate of the model which lies between 0 and 1. This is used to determine the extent to which new information can override old information. Q-Learning in this case has been used to estimate the reward values of the state in real-time. This is done due to two factors; 1) it is possible to estimate the value function in situations where a model doesn't exist, and 2) it can converge to an optimal policy. In algorithm 2 we represent the reinforcement learning and how it integrates into a Markovian decision.

Algorithm 2. RL integrated with Markovian Decision process

Input: A number of states, A number of actions, Reward values for all the states of the model

O**utput:** Updated model

1. t=0

2. $S_t \in$ S

3. repeat

4. Choose an action, a $\in A(S)$

5. Perform an action

6. observe the new state, S_{t+1} and received reward R(S_t)

7. Update the value function, V(S_t),

8. Update the Reward value for the current state, R(S_t) (discussed in algorithm 4)

9. Update the current state to the new state, S_{t+1}

10. t=t+1

11. Until there is no request from the user.

In the below equation (5), the Q-value function has been used in the proposed approach to calculate the value of the reward at state, S_t, which is called $\rho(S_t)$. It can be written as:

$$\rho(S_t) = 1 - \text{similarity}(\text{CrawlResultsA}, \text{CrawlResultsB}) + \gamma \max \rho(S_{t+1}) \quad (5)$$

Up to this point, we have leant that the reward is calculated automatically and updated incrementally during the model generation process. As a result, the following properties apply;

- The current state S_t is assigned to the URL_A and the next state S_{t+1} is assigned to the URL_B. Crawl, this simply fetches the current page (A or B), is called for both URLs and converts them into two strings, the words and links on each page.

- A similarity method is used to calculate the difference between the two URLs' content (strings). This is achieved using the Levenshtein distance. That is, when a user sends a new request, the algorithm counts the Q-value of upcoming states (web pages) and chooses the maximum amount to learn the reward of the current state.
- To eliminate redundancy in the model, the merging step, gkTail inference algorithm [34], is applied. As the name suggests it is used to merge the equivalent states. This means that if two states share the same label, rewards and immediate futures, they are merged. This typically means that their adjacent states also have the same labels and reward values. This procedure stops the system from redundant states with the same values.
- As discussed above, in the model generation process, all reward values are calculated incrementally. If a page is dynamically created using AJAX requests, then in our system it is also possible to measure the reward value. This was not possible by Emam et al. [25] proposed system as they only utilize the server-side log files. The overall automatic reward calculation procedure is represented by algorithms 3, 4 and 5.

Algorithm 3. Automatic reward calculation algorithm

Input: Model states (the URLs that already included in the model), S_t (the URL of the web page where user view currently), S_{t+1}(the URL of the webpage user choose from current page) , t=0,

O**utput:** Reward values, R for the all the states of the model

1. For each state S_0 to S_N do

2. Find the Crawl results for S_t and S_{t+1}

3. $\rho(S_t) = 1 - \text{similarity}(\text{Crawl}(S_t), \text{Crawl}(S_{t+1}))$

4. max=max $\rho\ \gamma((S_{t+1}))$

5. $\rho(S_t)$=($\rho(S_t)$+ .9 max)/100

6. Repeat Until no new $S_t \in S$ is found.

Algorithm 4. Similarity score measurement algorithm

Input: Method calls in String format (C_1, C_2)

Output: Similarity score

1. Similarity (C_1, C_2) **begin**
2. **If** (Length. C_1 < Length. C_2)
3. **then** swap (C_1, C_2)
4. BigLength ← Length. C_1
5. **Return** (BigLength-ComputeEditDistance(C_1, C_2))/BigLength

Algorithm 5. Merging the redundant states of the model

Input: The model states, S_i(state that comes new in the model), S_j(the set of states which are already labeled by the reward values)

Output: New merged model

1. for each (S_i, S_j) do
2. merge (S_i, S_j) if
3. (S_i= S_j) And (Reward (S_{i+1})= Reward (S_{j+1})) and (Adjacent (S_i)= Adjacent (S_j)) and ((Reward (Adjacent (S_i)) = Reward (Adjacent (S_j)))).
4. Repeat until no $S_i \in S$ is found

3.3.4 Resolving ambiguities in the models to generate accurate suggestions and next steps for the user

PRISM [2018], a probabilistic model checker, is used in our proposed system to evaluate the properties of the generated model; this is done to accurately analyze the behavioral model that has been inferred in our system. This is important in finding out the set of DTMCs which are more relevant to the specified properties.

This research is aimed at establishing the properties of the final model's specific reward value in the different states as they are used as suggestions to the user. The said properties are related to the *expected values* of the rewards. The $\mathcal{R}$ operator of PRISM is utilized in this process. The operator can be used either in a Boolean-valued query: $\mathcal{R}$ bound [rewardprop] or a real-valued query: $\mathcal{R}$ query [rewardprop], here *bound* takes the form < r, <= r, >r or >= r for an expression r and *query* is =? min=? or max=?

For instance, in a hypothetical system to consider the reward value of all the states up to the state labeled as" ONEcard", the following is used;

$$\{\}\mathcal{R} =? [F \text{ ONEcard}]$$

In the bracket {}, we have the option of either specifying the scope of the property for a defined user class or leaving it empty to not be limited to any specific scope.

We need to merge the DTMCs to find out the overall inference model of our system. The procedure of merging the different classes of DTLCs is:

- The union of the states of the input DTMCs consists of the set of states in the new DTMC.
- The law of total probability is used to calculate the transition probabilities in the new DTMC

$$P_T(S_i, S_j) = \sum_{1 \le k \le n} P_k(S_i, S_j) \times P_i(u_k)$$

where, $P_i(u_k)\ is$ the probability for a user that exited state S_i to belong to the user-class u_k.

- The labels in the input must be the same as the labels in the new DTMC.
- The reward values of the states of the new DTMC must be the same as the reward values in the input DTMC.

As indicated above, PRISM is used to evaluate the specified property for the final DTMC. By using considered reward properties, PRISM can evaluate the truth or falsity of a property as well as compute the reward functions. In other words, the results received by the system are a result of

property and the DTMC being passed to PRISM. The real-valued query: $\mathcal{R}$ query [rewardprop] and the reachability reward F are used by PRISM to generate suggestions for the user. The system can provide two different suggestions; 1) General suggestions analyzed from a specific link of the websites. 2) Specific or custom generated suggestions for different users; a user can get suggestions based on their browser and operating system. This has been elaborated below;

1) Based on our two cases, a user visits the link "Bear tracks" on the University of Alberta website. The user has two options;
   a) Users can get the overall suggestions, not limited to a specific scope. As an example, a number of suggestions can be made from the link "Bear tracks", which can be interpreted as;

$$\{\}\mathcal{R} =?\ [F\ Bear\ Track]$$

   The resultant URLs, with screenshots, are then displayed in the suggestion bar.

   b) Besides this, a user may prefer using a specific browser and operating system and need suggestions according to their operating system and browser. For instance, using "Bear Track" as an example, a user can use (Chrome and Windows) as the browser and operating system (case studies are shown in section 4.4). This can be interpreted as;

$$\{Chrome\ \&\&Windows\}\mathcal{R} =_{?}\ [F\ Bear\ Track]$$

   The resultant URLs, again with the screenshots, are displayed in the suggestion bar. In both cases note that the system makes suggestions after ordering the output reward values from high to low.

2) The default suggestion is the "Home" page in cases where the web page lacks an auto-generated suggestion.

The automatic suggestion generation procedure is presented in algorithm 6.

Algorithm 6. Automatic suggestions generation
Input: The user request, the updated model **Output:** A number of suggestions

1. Extract the Browser (B), Operating System (OS) and requested URL of the users, Suggestion=NULL

2. **if** the user selects the overall suggestion **then**

3. Suggestion= label ($\{\{\ \}\mathcal{R} =?\ [F\ RequestedURL]\}$

4. **else**

5. Suggestion= label ($\{\{B\ \&\&\ OS\}\mathcal{R} =?\ [F\ RequestedURL]\}$

6. **end if**

7. **if** Suggestion==Null **then**

8. print the Suggestion with the screenshot of the requested URL

9. **else**

10. print the "Home" page of the website as a suggestion with the screenshot of that

11. end if

3.4 EVALUATION and VALIDATION

In this section, we include an evaluation and validation of our system. We start the section with a brief description of the datasets used in this endeavor. Then we present the user study. After that, we present case studies to demonstrate how users of the website have benefitted from our system. Next, we represent the evaluation of the prediction results produced by our proposal; and finally, conclude our evaluation with a presentation of cross-validation of the results.

3.4.1 Clickstream Dataset

We utilize 5 datasets from two different types of websites; we use server log files from the University of Alberta[1] website which has two types of datasets; and RUETOJ[2] which has three types of datasets. We do not use any personal information from the users for ethical anonymity.

These static viewpoints of the system are a less than perfect scenario, however, a continuous long-term evaluation is not available at this juncture.

Figures 3.1 and 3.2 are used to visualize the 5 types of datasets, where we only consider the main links. The percentage of visits at the "University of Alberta" datasets shows a high percentage of visits at "Bear Tracks" and "Home". Links such as "ONEcard"," Student Services", "Login", "Search", "Menu" and "Email & Apps" have around 6- 8 % of visits. Links such as "Photos", "Videos", "Athletic" and "Transit" have very few visitors in both data sets. Figure 3.2, the RUET OJ data sets, shows that there are a few links that make up the major share of the number of visitors; "Profile", "Set Contest Problem", "Description of Problem", "Login" and "Standings" make up to around 60% percentage of visits. The remaining percentages are not evenly distributed across the other links of the website, links such as "Signup" and "Debug" have less than 1% of visits.

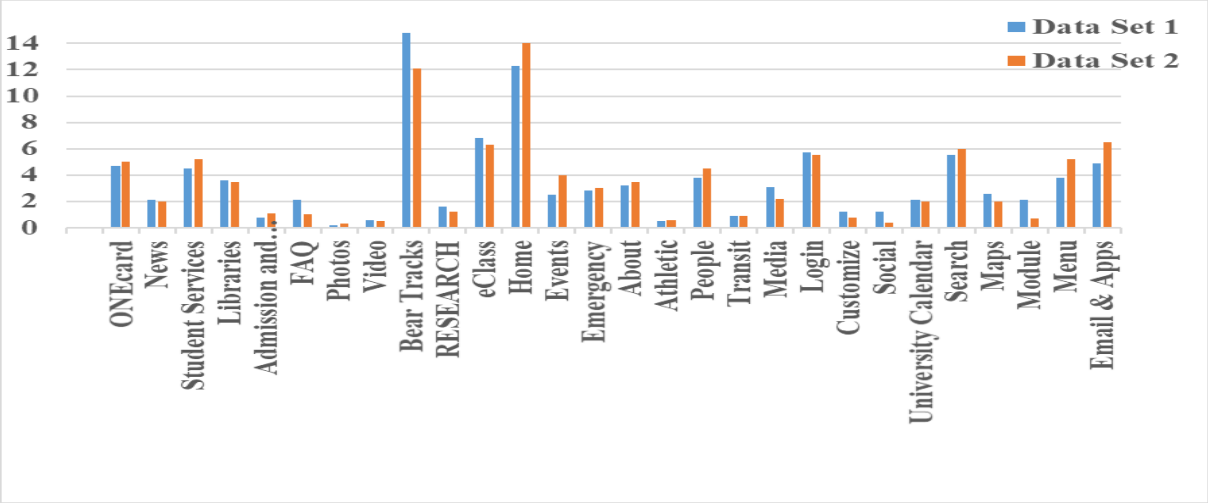


Figure 3. 1 Data set visualization for "University of Alberta" website

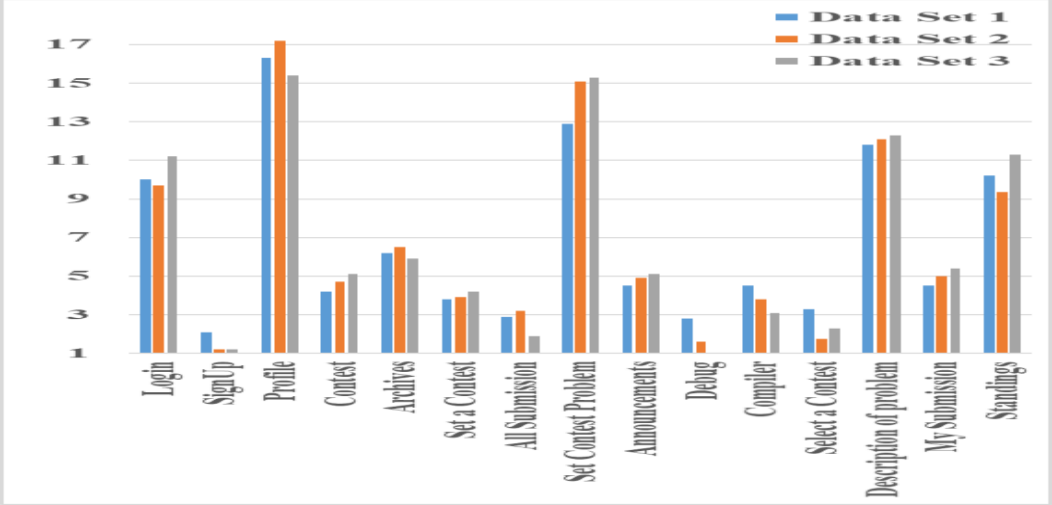


Figure 3. 2 Data set visualization for "RUET Online Judge" website

3.4.2 User Study

We have conducted a user study to gauge initial user reaction to utilizing such a navigation-assistance system; we asked users five simple questions.

- Q1. Does the system decrease the searching time of a web page?
- Q2. The suggestions are helpful?
- Q3. The system can give an overall idea about a webpage without visiting that page?
- Q4. The prediction of the next visited links is accurate?
- Q5. Overall, the system is improving my navigation experience?

Users answered questions 1 to 5 where 1 means strongly disagree and 5 means strongly agree. We collected data for these questions once when a user visits the website without using our add-on (conventional system) and once after using our developed add-on in their web browser (proposed system). Then we determine the Group Frequency Distribution, GFD using the equation:

$$GFD = L + \frac{\frac{N}{2} - Cfl}{F_m} \times R_W \qquad (8)$$

Where L is the lower-class boundary of the median; N is the number of people who take part in this study; Cfl is the cumulative frequency of the groups before the median group; F_m is the frequency of the median group; and R_W is the width of the group range. Table 3.1 summarizes the findings on the factors for "UofA" (40 users) and Table 3.2 for "RUET OJ" (30 Users).

Table 3. 1 Test analysis("UofA") for Factors Data conventional system and our proposed system

Quarters	Q1	Q2	Q3	Q4	Q5
Conventional system	GFD=3.3	GFD=3.8	GFD=3.4	GFD=3.9	GFD=3.3
Proposed system	GFD=4.6	GFD=4.4	GFD=4.8	GFD=4.4	GFD=4.8
U-Test	0.007937	0.01587	0.00653	0.01529	0.00521
Cliff's Delta	1	0.92	1	0.9	1

Table 3. 2 Test analysis ("RUET OJ") for Factors Data conventional and our proposed system

Quarters	Q1	Q2	Q3	Q4	Q5
Conventional system	GFD=3.2	GFD=3.9	GFD=3.3	GFD=4.0	GFD=3.1
Proposed system	GFD=4.9	GFD=4.6	GFD=4.9	GFD=4.6	GFD=4.9
U-Test	0.00525	0.0092	0.00325	0.0089	0.00243
Cliff's Delta	1	0.96	1	0.95	1

From the test analysis, we observe that there is a statistically significant difference in user experience between the conventional system and our proposed system. The result of GFD, U-test and Cliff's Delta indicates that our proposed system can provide a very helpful suggestion that can

improve the navigation experience. This usability study can be considered as a proxy for "real-time behavior", as the users experience the system instantly and over an extended period.

3.4.3 Case Studies (Different types of users in the web system)

Next, we present an in-depth analysis of users of the two websites: "The University of Alberta" and "RUET Online judge". Due to lack of space, we focus on three types of users for each website. In these case studies, we show how the user can be benefitted from our real-time system.

3.4.3.1 Case Study 1 ("Bear Track" users of University of Alberta website)

"Bear Tracks" is one of the important links on the Website. Students use this link to register or drop a course, see their grades etc. From figure 3.3(a) we see that our system suggests the "Bear Track" link as it is one of the most visited links and it shows that login is needed to visit that page. After the login three suggestions are displayed, a user can search a class for registration, check the class schedule and check their grades (figure 3.3(b)). If a user selects the "Search class" option, then there are two suggestions: modify the search and add that class to his class list, figure 3.3(c).

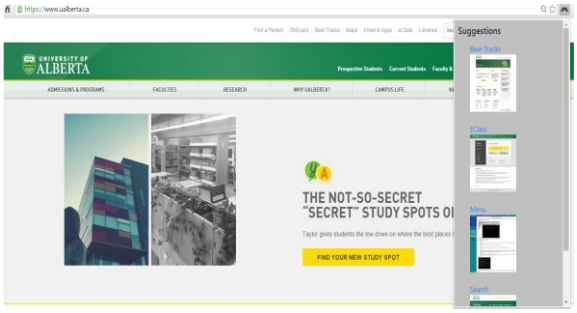
3.4.3.2 Case Study 2 ("Library" users of University of Alberta website)

There are many useful links on the Library pages including an online chat system. Figure 3.3(d) shows our automated system suggesting a library user chat with a representative, the link to the advanced search option on the library database and the other library services. If users go to the library services option, then according to figure 3.3(e) they can see the most popular service links provided by the University of Alberta.

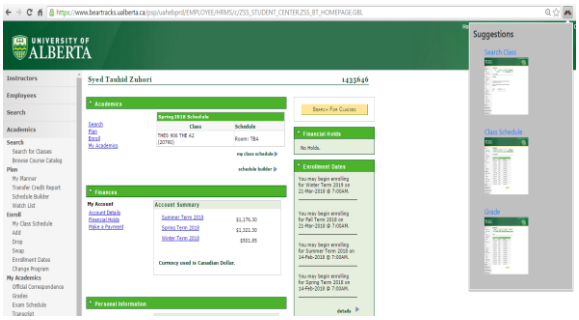
3.4.3.3 Case Study 3 ("One card" users of University of Alberta website)

"One card" is the ID card at the University of Alberta. From figure 3.3(f) we can see that users are suggested to go to the link to get one card or manage one card. Users can visit the main

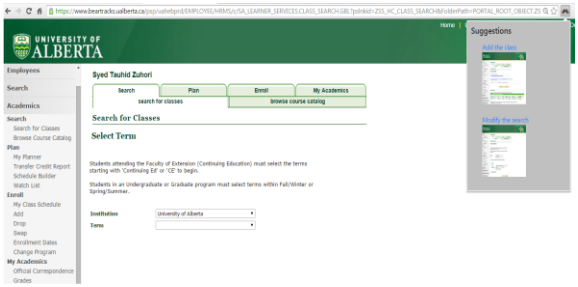
links “Email & Apps”, “Student Services” etc. If a user wants to get one card, then he is suggested to apply for it, figure 3.3(g). On the other hand, if a user already has one card then they can choose the “Manage One card” option. Figure 3.3(h) shows that there are three suggestions for that option: check account balance, manage meal plans and deposit funds at “One card”.



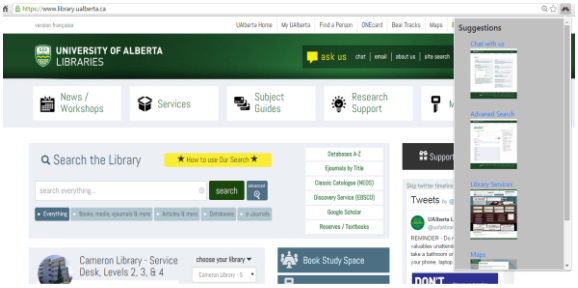
a) Home page



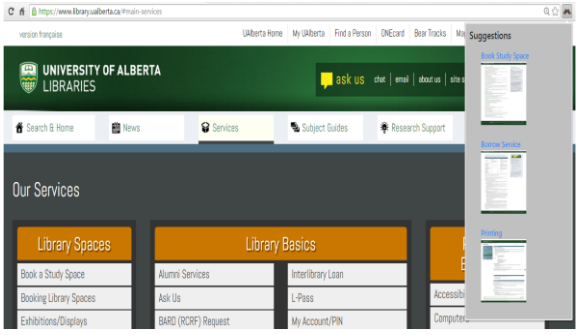
b) Bear Tracks page



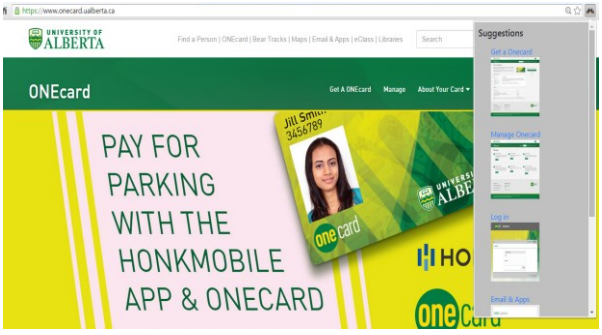
c) Search the course page



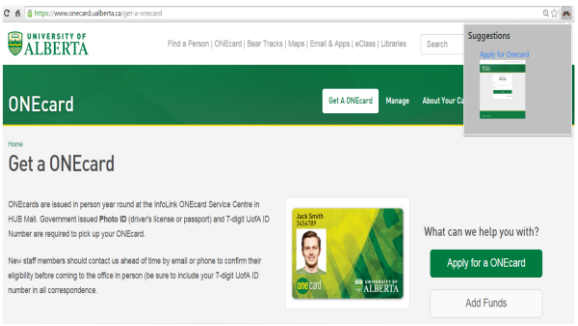
d) Libraries page



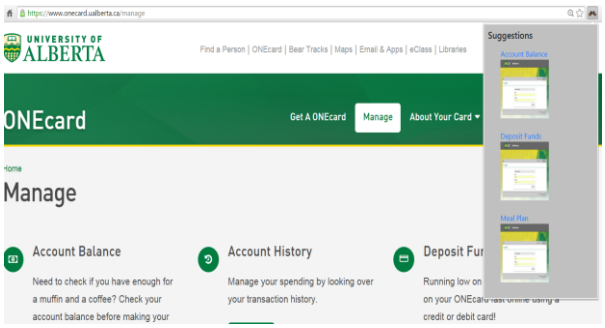
e) Library services page



f) ONE card page



g) Get a one card page



h) manage one card page

Figure 3. 3 Screenshot of our automated system suggestions for the University of Alberta website

3.4.3.4 Case Study 4 ("Judge" of the RUET Online Judge website)

A judge can create a contest and evaluate the code of a contestant, Figure 3.4(a). After a login, figure 3.4(b), shows there are three suggestions; users can go to their profile, they can go to the contest, or can visit the problem archives. Figure 3.4(c) shows that a judge can "Set a contest" by choosing a suggestion. After that he can "Select contest problem", figure 3.4(d), and "announce the contest", figure 3.4(e).

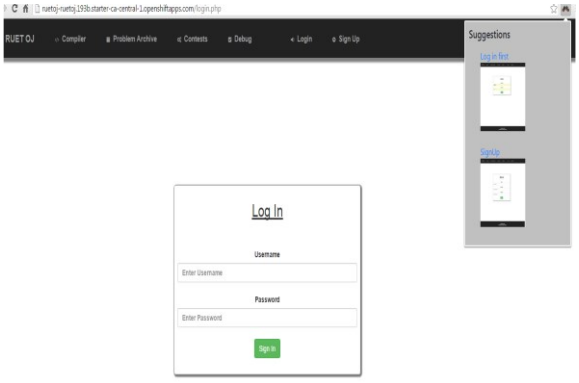
3.4.3.5 Case Study 5 ("Contestant" of the RUET Online Judge website)

Contestants can take part in any contest. They login and choose the suggestion "Contest", figure 3.4(b). Then they can choose a live contest, Figure 3.4(f) shows that page where they have two suggestions from our automated system: "Your Submission" and "Standing". So, they can submit the problem in time and check their position in the contest.

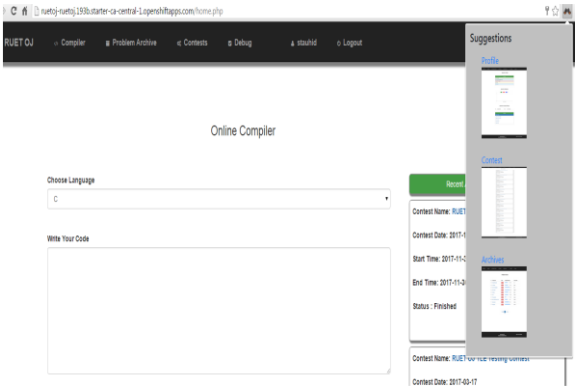
3.4.3.6 Case Study 6 ("Problem Solver" of the RUET Online Judge website)

The majority of users of RUET OJ are included in this category; they can choose "Archives", figure 3.4(b). Our system shows them the most visited problems as suggestions; figure

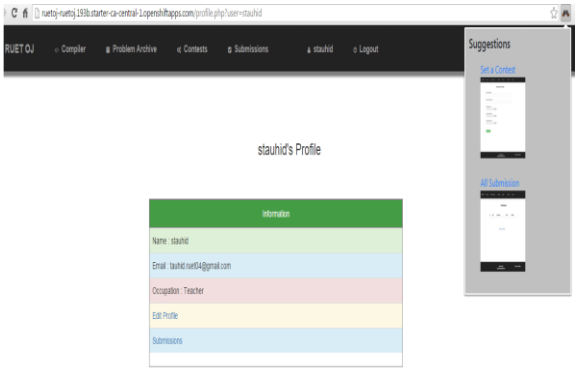
3.4(g) shows that. Figure 3.4(h) shows that there are two suggestions for the problem solver: users can check all their submissions, or they can return to their profile.



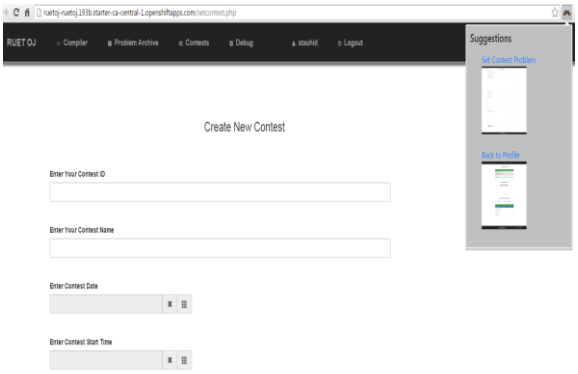
a) Home page



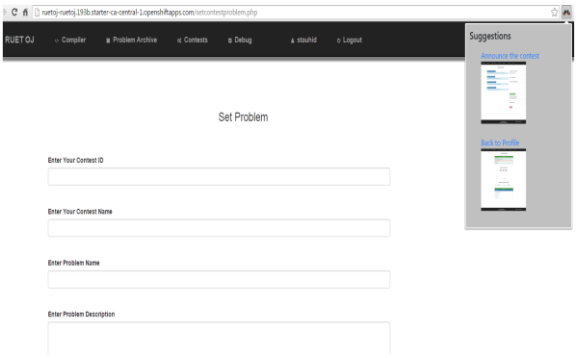
b) Users page



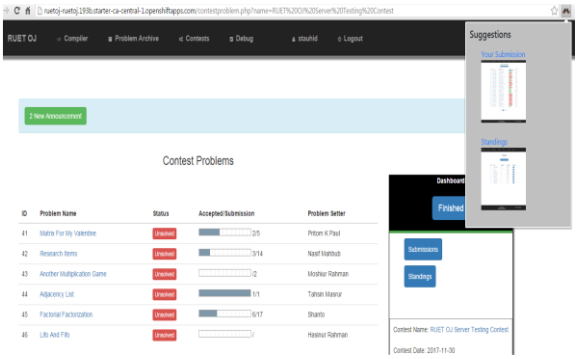
c) Users profile page



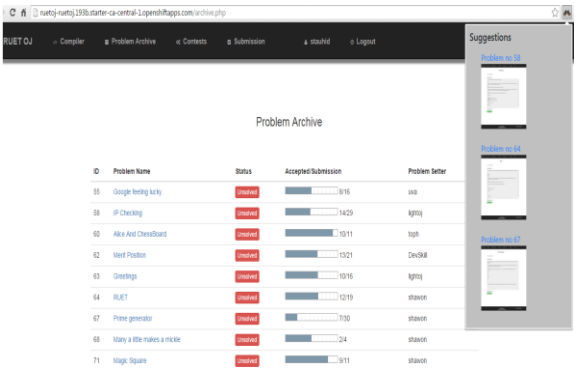
d) set a contest page



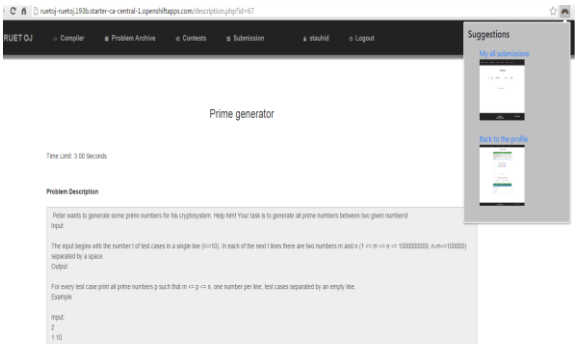
e) setting problem page



f) Contest page



g) Archives page



h) description of the problem page

Figure 3. 4 Screenshot of our automated system suggestions for RUET OJ website

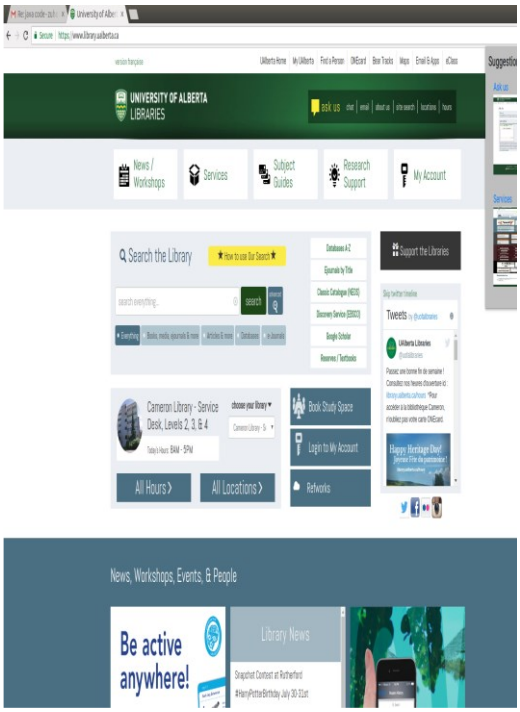
3.4.4 Case Studies (different browsers and platforms)

Figure 3.5 shows the "library" page of www.ualberta.ca website on different browsers and platforms. **The different browsers used are** (Chrome, Mozilla and Opera) while the operating systems are (Windows, Linux and MAC). This has been used as a testament that our proposed system does generate different suggestions while used on different browsers and operating systems. This has been illustrated in table 3.3.

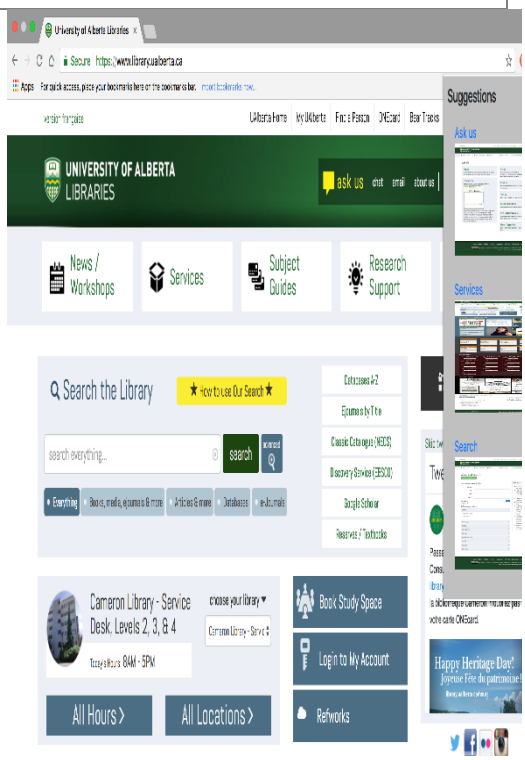
Table 3. 3 Suggestions for the different users from the "Library" page of "www.ualberta.ca"

Browser	Operating System	Suggestions
Chrome	Windows	1. "Ask us" 2. "Book study space"
Chrome	Linux	1. "Ask us" 2. "Services"
Chrome	MAC	1. "Ask us" 2. "Services" 3. "Search"

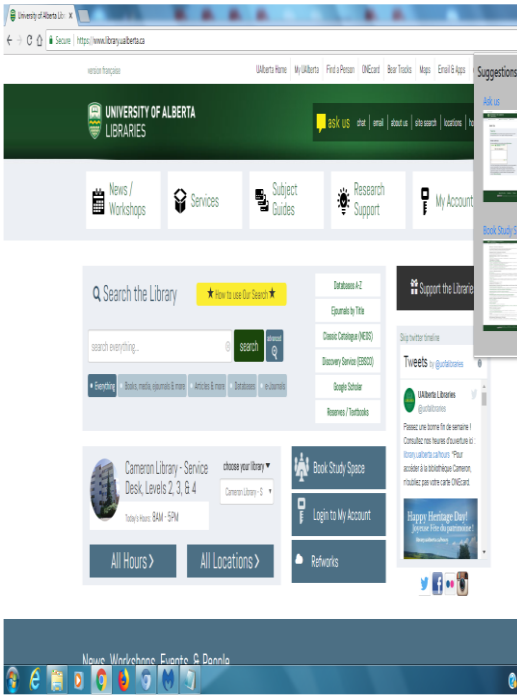
Mozilla	Windows	1. “Ask us” 2. “Services” 3. “Research support”
Mozilla	Linux	1. “Search” 2. “Book study space”
Mozilla	MAC	1. “Search” 2. “Services” 3. “Book study space” 4. “Research support”
Opera	Windows	1. “Ask us” 2. “Search”
Opera	MAC	1. “Ask us” 2. “Subject Guides”



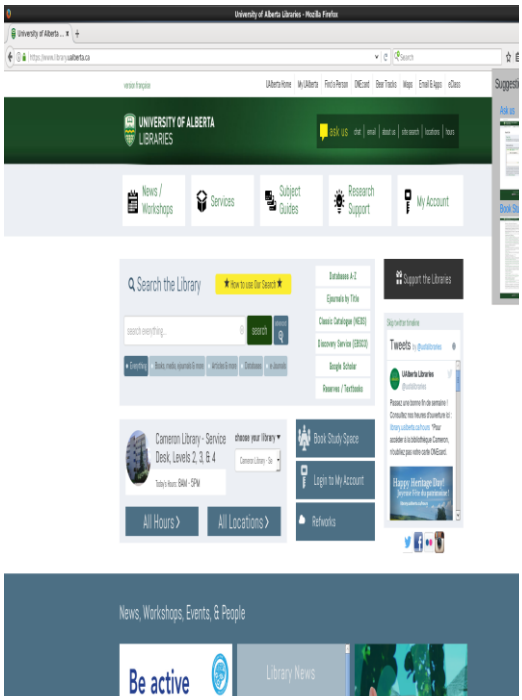
(a) Chrome in Linux



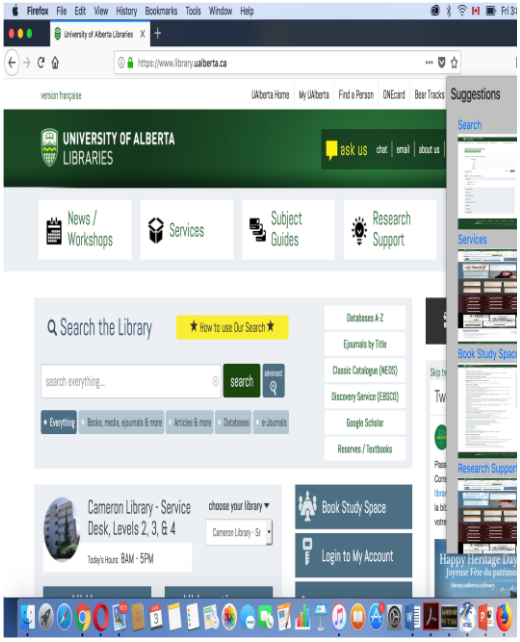
(b) Chrome in MAC



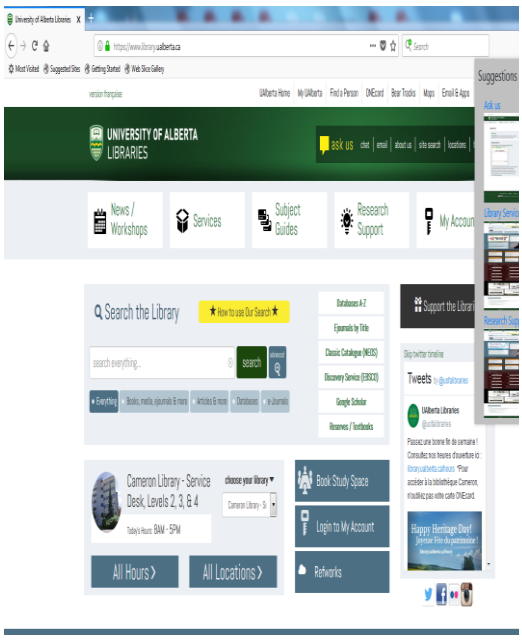
(c) Chrome in Windows



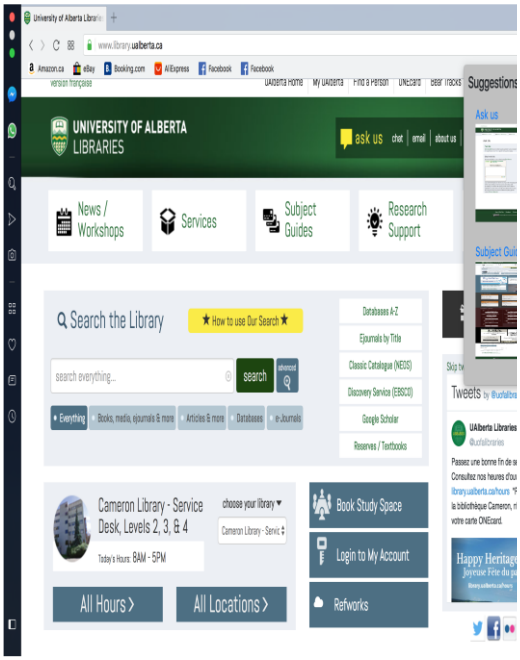
(d) Mozilla in Linux



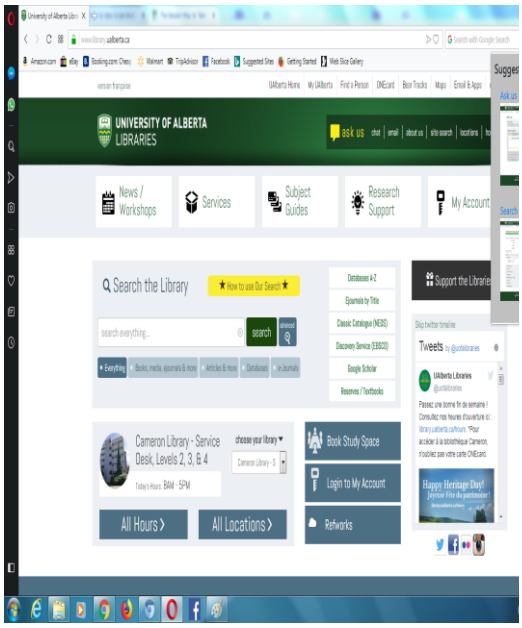
(e) Mozilla in MAC



(f) Mozilla in Windows



(g) Opera in MAC



(h) Opera in Windows

Figure 3. 5 Screenshot of "https://www.library.ualberta.ca/" in the nine browsers scenarios

3.4.5 Evaluation of Results

For finding the prediction accuracy, we first use the Dataset 1 of the University of Alberta as the training set and use the second dataset as the testing set. For the case of RUET OJ, we use the first dataset to train the model and use the combination of the second and third as the testing set.

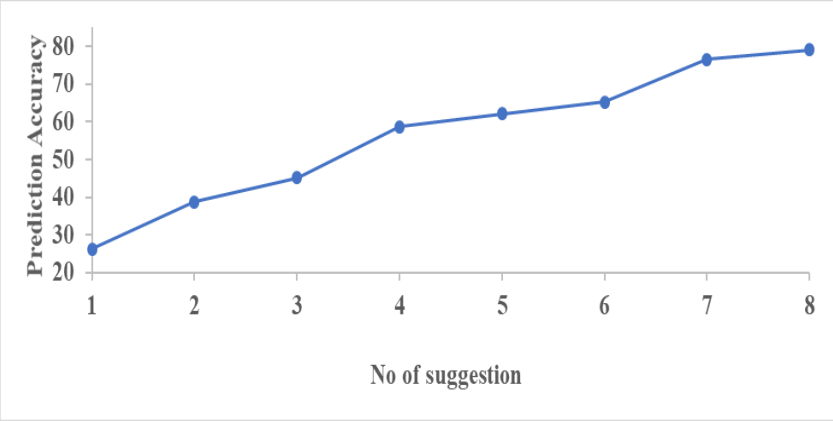


Figure 3. 6 Number of suggestions vs Prediction accuracy (UofA)

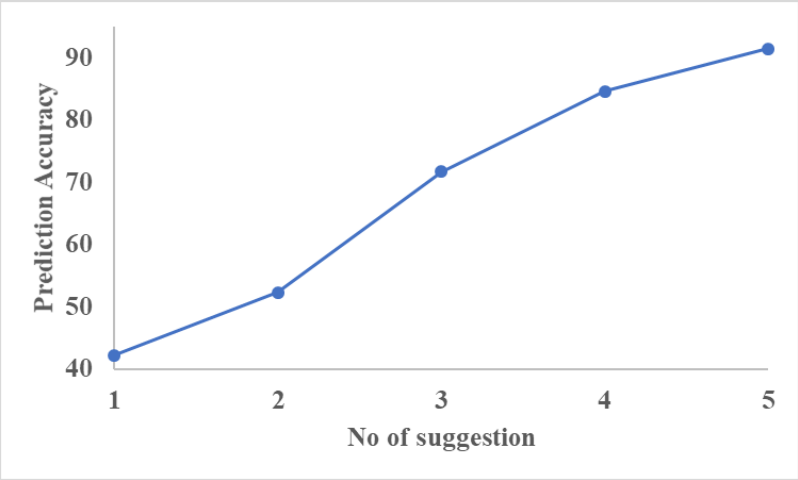


Figure 3. 7 Number of suggestions vs Prediction accuracy (RUET OJ)

Figure 3.6 represents the prediction accuracy with respect to the Number of suggestions for the University of Alberta website. As indicated in figure 3.6, with the maximal number of suggestions increasing from 1 to 8, the accuracy also increases. When only one web page is suggested, the accuracy is at its lowest, with a value of about 26.5%. This is a gradual rise to 79.14% when the number of suggestions is 8. From figure 3.7 (for RUET OJ system), the number of suggestions started at 1 which is 42% to 5 suggestions where the accuracy is at 91.46%.

Figure 3.8 includes values that have been collected from the University of Alberta website. The test involves recording the number of times a user clicked on a web page divided by the number of times our proposed system suggested that the user clicks on that page. The data indicates, that out of the 28 pages that are used for the test case, the most predicated based on accuracy, is the “Home” with a 92% and the “Login” with 86%. This makes sense since most users visit those pages to enable navigation to other web pages. Similar to figure 3.8, figure 3.9 is carried out on the RUET OJ website. The data indicates in this case that the “Login”, “Set a contest” and the “Signup” page has the highest mark. The system indicates a relatively low mark of 72% on the “submission” page.

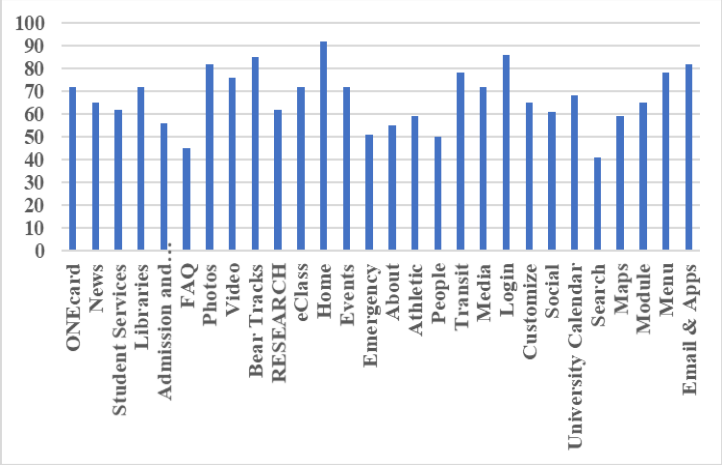


Figure 3. 8 Unique links vs Prediction accuracy (UofA)

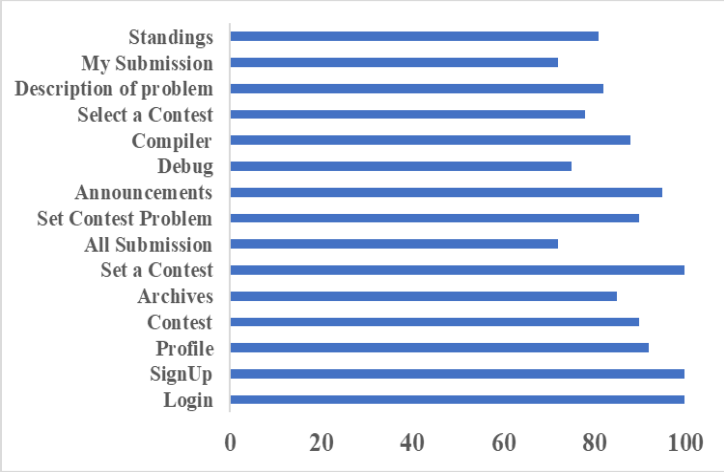


Figure 3. 9 Unique links vs Prediction accuracy (RUET OJ)

After that, we test how well our proposed system could make the predictions of the users' next visited links. Therefore, certain indicators needed to be defined in advance to evaluate the performance of the system. The indicators used in our system are accuracy, precision and recall, three well-accepted performance indicators in the information retrieval field.

Figure 3.10 represents the prediction accuracy with respect to the percentage of users for the “UofA” dataset. The test cases are carried out incrementally; it starts with one percent of a select total number of users thereby testing the accuracy of the prediction. This is then repeated for 2%, 3% incrementally to 100%. Except for the last group, where the whole population is considered, 10 different (random) variations of the groups are considered and the mean value of

the performance is considered for evaluation purposes. We consider two cases; in the first case, dataset 1 is considered as training and dataset 2 as testing. We consider the opposite in case 2. From figure 10 we observe that in case 1, the system stabilizes at around 80.69% and the accuracy lies between 74% and 81%. For case 2, the accuracy lies between 75% and 82%. Figure 3.11 represents the accuracy for "RUET OJ" datasets where we consider dataset 1 as a training set and a combination of datasets 2 and 3 as a testing set in case 1. In case 2, we consider the opposite of case 1. From figure 3.11, we observe that in case 1 the system stabilizes at 88.5% and in case 2 at 85.8%.

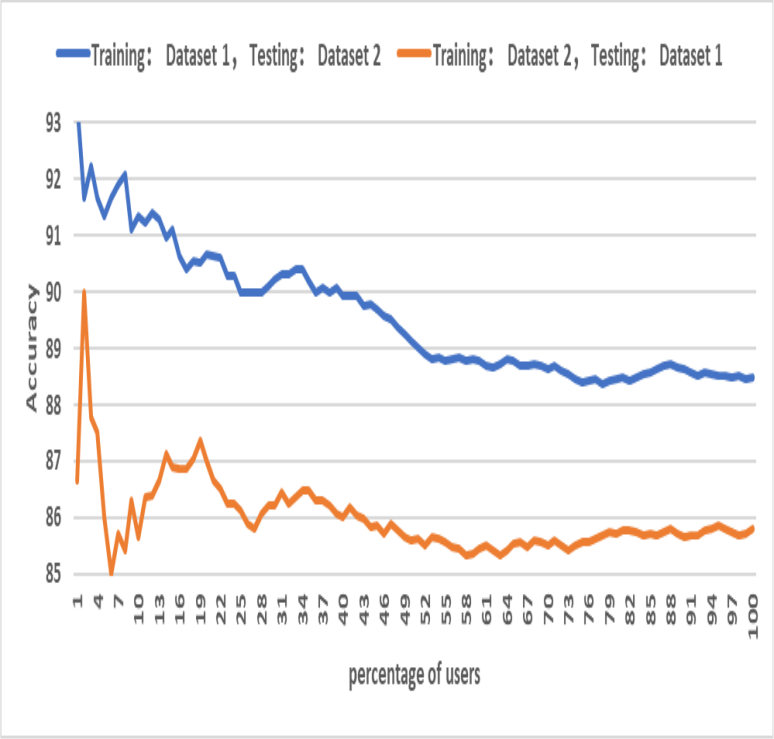


Figure 3. 10 Accuracy in "UofA"

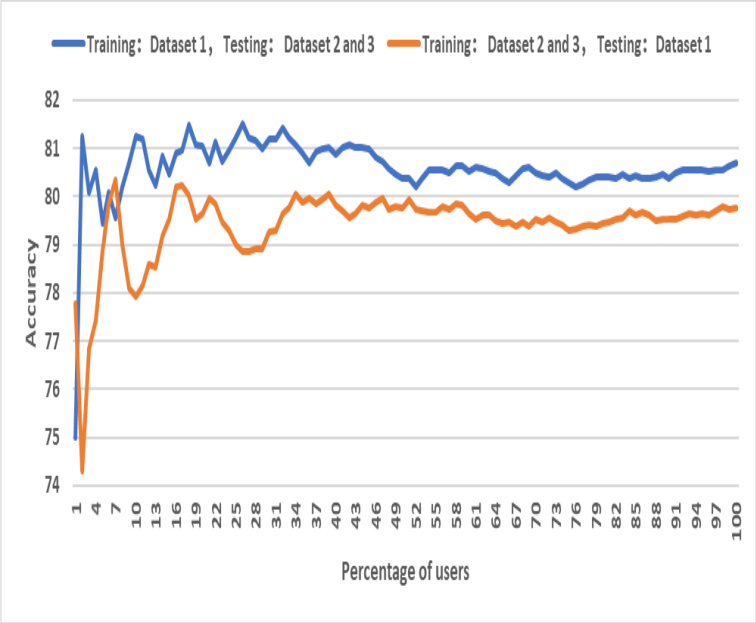


Figure 3. 11 Accuracy in "RUET OJ"

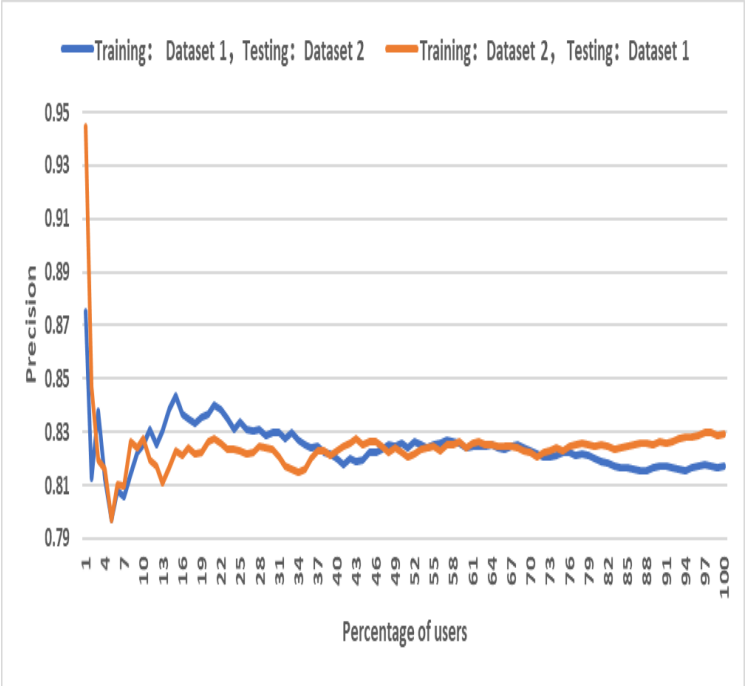


Figure 3. 12 Precision in "UofA"

Figure 3. 13 Precision in "RUET OJ"

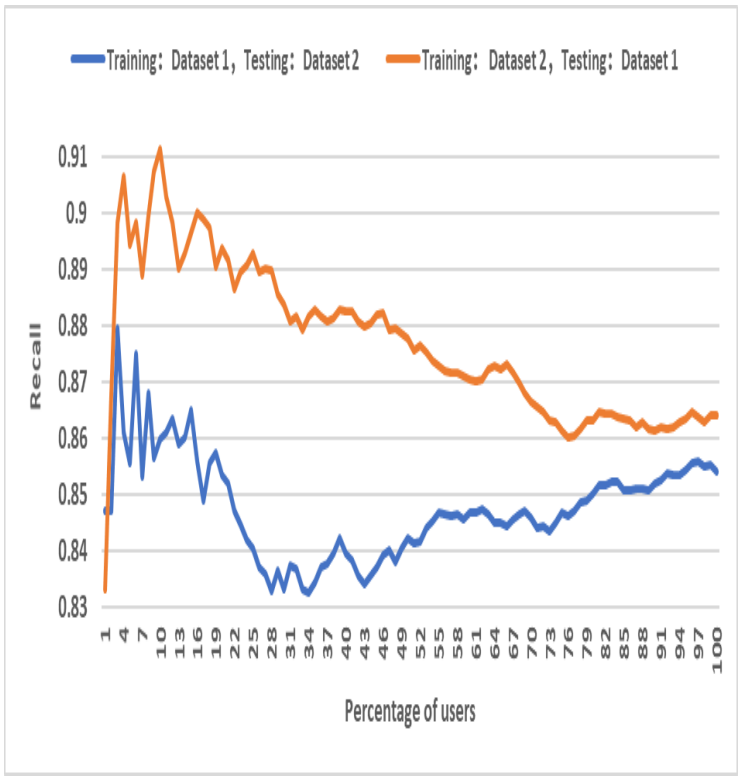


Figure 3. 14 Recall in "UofA"

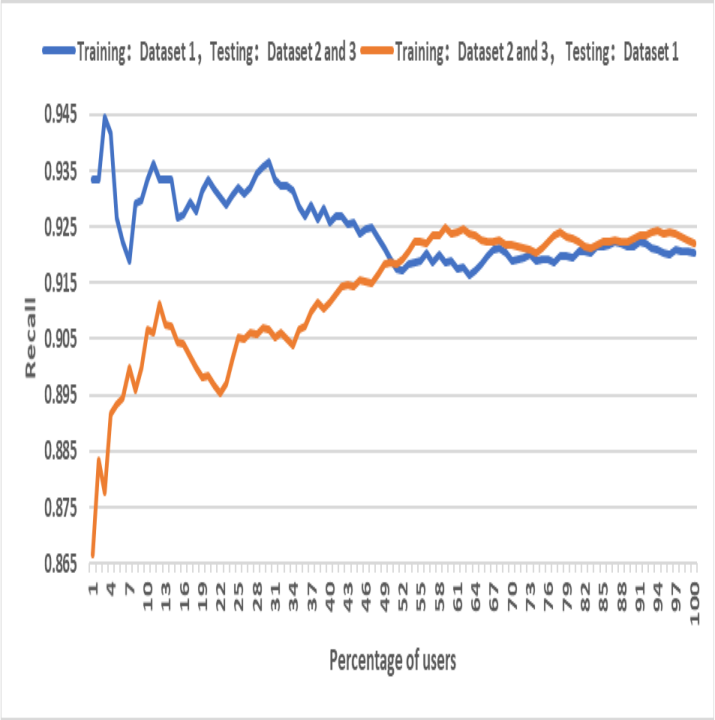


Figure 3. 15 Recall in "RUET OJ"

Then, we find the precision and recall. For example, if a visitor of the system browsed several pages on the website and our proposed system suggested six other web pages for browsing, of which four pages appeared in the actual visiting history, then the precision of the system is 4/6 or 0.66. If the relevant page number of the current user was five, then the recall of the system would be 4/5 or 0.8. Figure 3.12 represents the precision of "UofA" datasets where the average precision for case 1 is 0.817 and case 2 is 0.829. Figure 3.13 represents the precision of "RUET OJ" datasets where the precision lies between 0.88 and 0.93 for case 1, and 0.91 to 0.932 for case 2. In the case of a recall, figure 3.14 represents that for the "UofA" dataset. the average recall is 0.854 for case 1 and 0.865 for case 2. On the other hand, in the "RUET OJ" datasets, the system stabilizes at a recall value of 0.920 in case 1 and 0.922 in case 2; see figure 3.15.

We do a simple statistical analysis for finding the difference between performance indicators for both cases of "UofA" and "RUET OJ". We used the Mann-Whitney U test for statistical analysis.

Table 3. 4 Mann-Whitney U test for comparing case 1 and case 2 (Value of $P(Z|H_0)$)

"UofA" system						"RUET OJ system"					
Accuracy		Precision		Recall		Accuracy		Precision		Recall	
Case 1	Case 2	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2
0.7937		0.6252		0.1587		0.3257		0.2785		0.2885	

The two cases are compared in pairs, and the hypothesis is that they have equal accuracy and set 0.05 as the significance level. Table 3.4 shows the value of the U-test. From the table, we observe that for all the cases, the U-test value is greater than the significant level. So, we cannot reject the null hypothesis that the accuracy, precision or recall of one case is the same as another case with a significance level of 5%. So, from this experiment, we can decide that, if we exchange our dataset during the training and testing period then the system performance will remain "similar".

3.5 CONCLUSION

This chapter introduces a real-time system to assist website users. To achieve this, our proposed system captures the requests submitted by web users to the webserver in real-time and generates an inference behavioral model using the Discrete Time Markov Chain inference process. After that, it produces navigation suggestions for the users and updates the inference model by using a Markovian Decision Process. For evaluating our system, we conduct a user study, and case studies on different types of users on two different websites and use three well-accepted performance indicators; accuracy, recall and precision. The results of the user study show that our proposed system significantly outperforms the conventional approaches. Besides this, the performance indicators seem to represent "acceptable" results and remain similar across a variety of investigations. Our research can automatically suggest web page URLs in real-time. Such a system can save time for visitors to websites, and lead to a better information service. Using our

proposed system, the website developer can also improve the website's hyperlink structure by better understanding and predicting users' navigational behavior.

Chapter 4: Summary of Conclusions

4.1 SUMMARY of THESIS

The purpose of this thesis is to assist website users to visit a website efficiently even if it is new to them. Web usage mining is used for that. In order to extract the usage patterns, we analyze the clickstream data. We face two major challenges; preprocessing the raw data to provide an accurate picture of how a website is used. The second one is to present the rules and patterns that are potentially interesting to the users by filtering the results. We build a novel real-time system that personalized browsing assistance based on website user requests. Our proposed system can provide accurate link suggestions with a live screenshot of the suggested web page. Therefore, users can see the content before visiting the web page. Moreover, our system can provide suggestions for different browsers and operating systems. We provide a user study, case studies and conduct experiments on five datasets to evaluate the system.

In chapter 2 we represent our project that develops a method to find the importance of web pages without using web browser data or invading the privacy of users. Rather, it works on the structure of a website. We propose a novel method that can take webpage content as input and produce a score for each page automatically. There are two important factors we consider; (1) "What is the minimum number of clicks needed to access web pages on a website?" and (2) "How a web page is linked with other web pages on a website?" We use CatBoost learning method to train our model by using the "web page views" results generated by "Google Analytics" and "SimilarWeb".

In chapter 3 we represent our journal paper. This work builds a personalized browsing assistant system. This is based on the current user request that the user submits to a web server. The behaviour model used in this system is developed based on a Discrete Time Markov Chain (DTMCs) inference process. That model can monitor the user activities in real-time and can suggest the next destination to the web user. Finally, it updates the model in real-time using a Markovian Decision Process (MDP).

4.2 PUBLICATIONS

1. Papers

- **Syed Tauhid Zuhori** and James Miller, **"Real-time Browsing Assistant on Web"**, IDAIS International Journal on WWW/Internet, Volume-16, Issue-2, Pages 1-18.
- **Syed Tauhid Zuhori** and James Miller, **"An Unsupervised Approach for User Behaviour Clustering of Websites using Navigation Patterns of the Web Users"**, Journal of Software Engineering & Intelligent Systems, Volume-3, Issue-2, Pages 174-191.
- **Syed Tauhid Zuhori** and James Miller, **"A New Approach for Web User Clustering Based on their Clickstream Similarity"**. (accepted by Journal of Software Engineering & Intelligent Systems).
- **Syed Tauhid Zuhori** and James Miller, **"A Novel Real-Time Browsing Assistance System Based on Web User Behaviours"**, 17th International Conference on WWW/ Internet, 21-23 October, 2018, Hungary.
- **Syed Tauhid Zuhori** and James Miller, **"Clickstream Clustering via User Similarity Estimation"**, 17th International Conference on WWW/ Internet, 21-23 October, 2018, Hungary.
- **Syed Tauhid Zuhori** and James Miller, **"A Model-Based Approach for Web User Clustering and Behaviour Analysis"**, 17th International Conference on WWW/ Internet, 21-23 October, 2018, Hungary.
- **Syed Tauhid Zuhori** and James Miller, **"An Automated Web Structure-Based Method for Predicting the Importance of a Webpage"**. (Accepted by London Journal of Engineering Research in Volume 22 Issue 2).

2. Posters and Presentations

- **Syed Tauhid Zuhori,** Abhimanyu Panwar and James Miller, "A New Methodology for User Behavior Profiling of Websites, 24th Annual International Conference on Computer Science and Software Engineering.

- **Syed Tauhid Zuhori** and James Miller, **"An Unsupervised Approach for Web User Behaviour Clustering using anonymous data",** 25th Annual International Conference on Computer Science and Software Engineering.
- Zhen Xu, James Miller and **Syed Tauhid Zuhori**, "A New Web Page Classification Model based on Visual Information using Gestalt Laws of Grouping", 25th Annual International Conference on Computer Science and Software Engineering.

Bibliography

1. Adeniyi, D., Wei, Z., Yongquan, Y.: Automated Web Usage Data Mining and Recommendation System using K-Nearest Neighbor Classification Method. Applied Computing and Informatics, 12, pp 90-108, 2016.

2. Arapakis, I., Bai, X., Cambazoglku. B. B.: Impact of Response Latency on User Behavior in Web Search, In: Proceedings of the SIGIR Conference on Special Interest Group on Information Retrieval, 103-112, ACM (2014).

3. Arbelaitz, O., Gurrutxaga, I., Lojo, A., Muguerza, J., Perez, J., Perona, I.: Web usage and content mining to extract knowledge for modelling the uers of the Bidasoa Turismo Website and to Adapt it. Expert Systems with Application, 40, pp7478-7491, Elsevier (2013).

4. Banerjee, A., Joydeep, G.: Clickstream Clustering using Weighted Longest Common Subsequences. In: 16th European Conference and Artificial Intelligence. 1-8 (2009).

5. Ghezzi, C., Pezze, M., Sama, M., Tamburrelli, G.: Mining Behavioral Models from User-Intensive Web Applications. In the Proceedings of the ICSE International Conference on Software Engineering (ICSE, 14), 277-288, ACM (2014).

6. Guan, W., Gao, H., Yang, M., Li, Y., Ma, H., Qian, W., Cao, Z., Yang, X.: Analyzing User Behavior of the Micro-Blogging Website Sina Weibo during Hot Social Events. Statistical Mechanics and its Applications, 395, pp 340-351, Elsevier (2014).

7. Gurini, D., Gasparetti, F., Micarelli, A., Sansonetti, G.: Enhancement Social Recommendation with Sentiment Communities. WISE, pp 308-315, Springer (2015).

8. Hamasaki, M., Goto, M.: Songrium: A Music Browsing Assistance Service Based on Visualization on Massive Open Collaboration Within Music Content Creation Community. WikiSym, 1-10, ACM (2013).

9. Jafari, M., Sabzehi, S., Irani, A.: Applying Web Usage Mining Techniques to Design Effective Web Recommendation System: A Case Study, International Journal on Advances in Computer Science, 3(2), 78-90 (2014).

10. Javari, A., Jalali, M.: Cluster Based Collaborative Filtering for Sign Predictions in Social Networks with Positive and Negative Links. ACM Transactions on Intelligent Systems and Technology, 5(2), 1-19 (2014).

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

12. Li, X.: Data Processing in Web Usage Mining. In: 19th International Conference on Industrial Engineering and Management, pp 257-266, Springer (2013).

13. Nagy, I., Papanek, C.: User Behavior Analysis Based on Time Spent on Web Pages. Springer-Verlag Berlin Heidelberg, SCI 172. pp 117-135 (2009).

14. Pang, W., Wen, X., YuRong, X., Yu, Z.: Link Prediction in Social Network: The State-of-the-art. Science China, 58(2), Springer (2015).

15. Schur, M., Roth, A., Zeller, A.: Mining Behavior Models from Enterprise Web Applications. ESEC/FSC 422-432, ACM (2013).

16. Shahryary, S., Shahryary, M., Noor, M.: A Community Based Approach for Link Prediction in Signed Social Network, Scientific Programming, 1-10 (2015).

17. Tan, Y., Yu, K., Wu, X., Pan, D., Liu, Y.: Predicting App Usage Based on Link Prediction In User App-Bipartite Network. SmartCom, pp 191-205, Springer (2018).

18. Wang, Y., Dai, W., Yuan, Y.: Website browsing aid: A navigation graph-based recommendation system. Decision Support System, 45, pp 387-400, Elsevier (2008).

19. Wang, G., Tristan, K., Wilson, C., Zheng, H., Zhao, B. Y.: You Are How You Click: Clickstream Analysis for Sybil Detection. In: Proceedings of the 22nd USENIX Security Symposium, 241-255(2013).

20. Wang, G., Zhan, X., Tang, S., Zhen, H., Zhan, B.: Unsupervised Clickstream Clustering for User Behavior Analysis. In: Proceedings of the CHI Conference on Human Factors in Computing System, ACM (2016).

21. Wan, M., Li, L., Xiao, J.: CAS based clustering algorithm for web users. Nonlinear Dyn.: 61, 347-361 (2010).

22. Wan, M., Jonnson, A., Wang, C., Li, L., Yang, Y.: A Random Indexing Approach for Web User Clustering and Web Prefetching. Springer-Verlag Berlin Heidelberg, pp 40-52 (2012).

23. Zhu, J., Hong, J., Hughes, G.: Using Markov Chain for Link Prediction in Adaptive Web sites. Soft-Ware, pp 60-73, Springer (2002).

24. F. Chierichetti and R. Kumar, “Are Web Users Really Markovian ?,” in Proceedings of the 21st international conference on World Wide Web, 2012, pp. 609–618.

25. Emam, S., S., Miller, J.: Inferring Extended Probabilistic Finite State Automation Models from Software Executions. ACM Transactions on Software Engineering and Methodology, 27(1), June 2018.

26. R. W. White, P. Bailey, and L. Chen, “Predicting User Interests from Contextual Information,” in Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, 2009, pp. 363–370.

27. C. Szepesvári, Algorithms for Reinforcement Learning, vol. 4, no. 1. Morgan & Claypool Publishers, 2010.

28. M. Wiering and M. van Otterlo, Reinforcement Learning (State of the Art). Springer, 2012.

29. J. Jiang, X. Song, N. Yu, and C. Lin, “FoCUS : Learning to Crawl Web Forums,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 6, pp. 1293–1306, 2013.

30. L. C. Stuart, “User Modeling via Machine Learning and Rule- Based Reasoning to Understand and Predict Errors in Survey Systems,” 2013.

31. M. Virvou, C. Troussas, and E. Alepis, “Machine learning for user modeling in a multilingual learning system,” pp. 292–297, 2012.

32. D. Henriques, P. Zuliani, and E. M. Clarke, “Statistical Model Checking for Markov Decision Processes,” vol. 1041377, no. 1041377, 2012.

33. P. Shitole and M. A. Potey, “Survey of User Modeling Techniques with Specific Emphasis on Considering Demographic Attributes,” vol. 3, no. 12, pp. 1366–1370, 2014.

34. S. Lamprier, No.Baskiotis, T. Ziadi and L. M. Hillah: The CARE Platform for the Analysis of Behavioural Model Inference Techniques. Journal on Information and Software Technology, 2015, 60 (32-50).

35. C. Baier, L. D. Alfaro, V. Forejt and M. Kwiatkowska: Model Checking Probabilistic System. Handbook of Model Chceking, 2018, pp 963-999.

36. B. Abrahao, S. Soundarajan, J. Hopcroft, and R. Kleinberg (2012), On the separability of structural classes of communities, in Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, ACM, New York, pp. 624–632.

37. D. Aldous and J. A. Fill (2002), Reversible Markov Chains and Random Walks on Graphs, unfinished monograph; recompiled 2014, available online from http://www.stat.berkeley.edu/~aldous/RWG/book.html.

38. S. Allesina and M. Pascual (2009), Googling food webs: Can an eigenvector measure species importance for coextensions? PLoS Comput. Biol., 5, e1000494.

39. R. Andersen, F. Chung, and K. Lang (2006), Local graph partitioning using PageRank vectors, in Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, IEEE, pp. 475–486.

40. W. N. J. Anderson and T. D. Morley (1985), Eigenvalues of the Laplacian of a graph, Linear Multilinear Algebra, 18, pp. 141–145.

41. A. Arasu, J. Novak, A. Tomkins, and J. Tomlin (2002), PageRank computation and the structure of the web: Experiments and algorithms, in Proceedings of the 11th International Conference on the World Wide Web, Poster session. www2002.org/COROM/poster.173.pdf.

42. K. Avrachenkov, N. Litvak, and K. S. Pham (2007), Distribution of PageRank mass among principle components of the web, in Proceedings of the 5th Workshop on Algorithms and Models for the Web Graph (WAW2007).

43. A. Bonato and F. C. Graham, eds., Lecture Notes in Comput. Sci. 4863, Springer, New York, pp. 16–28.

44. K. Avrachenkov, B. Ribeiro, and D. Towsley (2010), Improving random walk estimation accuracy with uniform restarts, in Algorithms and Models for the Web-Graph, R. Kumar and D. Sivakumar, eds., Lecture Notes in Comput. Sci. 6516, Springer, Berlin, Heidelberg, pp. 98–109.

45. L. Backstrom and J. Leskovec (2011), Supervised random walks: Predicting and recommending links in social networks, in Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM ’11, ACM, New York, pp. 635–644.

46. R. Baeza-Yates, P. Boldi, and C. Castillo (2006), Generalizing PageRank: Damping functions for link-based ranking algorithms, in Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR2006), Seattle, WA, ACM, New York, pp. 308–315.

47. B. Bahmani, A. Chowdhury, and A. Goel (2010), Fast incremental and personalized PageRank, Proc. VLDB Endow.,, pp. 173–184.

48. A. Balmin, V. Hristidis, and Y. Papakonstantinou (2004), ObjectRank: Authority-based keyword search in databases, in Proceedings of the Thirtieth International Conference on Very Large Data Bases, Volume 30, VLDB ’04, VLDB Endowment, pp. 564–575.

49. Z. Bar-Yossef and L.-T. Mashiach (2008), Local approximation of PageRank and reverse Page-Rank, in CIKM ’08: Proceedings of the 17th ACM conference on Information and Knowledge Management, ACM, New York, pp. 279–288.

50. D. S. Bassett and E. Bullmore (2006), Small-world brain networks, The Neuroscientist, 12, pp. 512–523.

51. M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang (2013), Message-passing algorithms for sparse network alignment, ACM Trans. Knowledge Discovery. Data, 7, pp. 3:1–3:31.

52. L. Becchetti, C. Castillo, D. Donato, R. Baeza-Yates, and S. Leonardi (2008), Link analysis for web spam detection, ACM Trans. Web, 2, pp. 1–42.

53. M. Benzi, E. Estrada, and C. Klymko (2013), Ranking hubs and authorities using matrix functions.

54. P. Berkhin (2005), A survey on PageRank computing, Internet Math., 2, pp. 73–120.

55. A. Berman and R. J. Plemmons (1994), Nonnegative Matrices in the Mathematical Sciences, Classics Appl. Math. 9, SIAM, Philadelphia.

56. M. Bianchini, M. Gori, and F. Scarselli (2005), Inside PageRank, ACM Trans. Internet Technologies, 5, pp. 92–128.

57. D. A. Bini, G. M. D. Corso, and F. Romani (2010), A combined approach for evaluating papers, authors and scientific journals, J. Computer. Applied Mathematics, 234, pp. 3104–3121.

58. D. M. Blei, A. Y. Ng, and M. I. Jordan (2003), Latent Dirichlet allocation, J. Mach. Learn. Res., 3, pp. 993–1022.

59. V. D. Blondel, A. Gajardo, M. Heymans, P. Senellart, and P. Van Dooren (2004), A measure of similarity between graph vertices: Applications to synonym extraction and web searching, SIAM Rev., 46, pp. 647–666.

60. P. Boldi (2005), TotalRank: Ranking without damping, in Poster Proceedings of the 14th International Conference on the World Wide Web (WWW2005), ACM Press, New York, pp. 898–899.

61. P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna (2008), The query-flow graph: Model and applications, in Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM ’08, ACM, New York, pp. 609–618.

62. P. Boldi, F. Bonchi, C. Castillo, and S. Vigna (2009a), Voting in social networks, in Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, ACM, New York, pp. 777–786.
63. P. Boldi, F. Bonchi, C. Castillo, and S. Vigna (2011), Viscous democracy for social networks, Commun. ACM, 54, pp. 129–137.

64. P. Boldi, R. Posenato, M. Santini, and S. Vigna (2007), Traps and pitfalls of topic-biased PageRank, in Fourth International Workshop on Algorithms and Models for the Web-Graph, WAW2006, Lecture Notes in Comput. Sci., Springer-Verlag, New York, pp. 107–116.

65. P. Boldi, M. Santini, and S. Vigna (2005), PageRank as a function of the damping factor, in Proceedings of the 14th International Conference on the World WideWeb (WWW2005), Chiba, Japan, ACM Press, New York, pp. 557–566.

66. P. Boldi, M. Santini, and S. Vigna (2009), PageRank: Functional dependencies, ACM Trans. Inf. Syst., 27, pp. 1–23.

67. J. Bollen, M. A. Rodriquez, and H. Van de Sompel (2006), Journal status, Scientometrics, pp. 669–687.

68. S. B. Brawer, M. Ibel, R. M. Keller, M. Shivakumar (2016), Web Crawler Scheduler that utilizes Sitemaps from Websites, United States Patent.

69. T. Friedrich, S. Schlauderer, S. Overhage (2019), The impact of social commerce feature richness on website stickiness through cognitive and affective factors: An experimental study, Electronic Commerce Research and Applications, 36(2019).

70. 1] Lundberg S. M., Lee Su-In. A Unified Approach to Interpreting Model Predictions. 2017. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

71. Lundberg S. M., Erion G. G., Lee Su-In. Consistent Individualized Feature Attribution for Tree Ensembles. 2017. https://arxiv.org/pdf/1802.03888.pdf

72. Khun S. W., Petzer D. J., Fostering Purchase Intentions Toward Online Retailer Websites in an Emerging Market: An S-O-R Perspective, Journal of Internet Commerce (2018), Volume 17 Issue 3, page 255-282.