IADIS International Journal on WWW/Internet
Vol. 16, No. 2, pp. 1-18
ISSN: 1645-7641

REAL-TIME BROWSING ASSISTANT ON WEB

Syed Tauhid Zuhori and James Miller
University of Alberta, Edmonton, Canada

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.

KEYWORDS

Real-time Browsing Assistance System, Discrete Time Markov Chain Inference Process, Markovian Decision Process, Reinforcement Learning

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 web-site. Our system will suggest to users their next navigation steps (hyperlink following) according to previous activity on the website from previous users.

Our automated system helps users to find the optimal page to be visited next. A good illustration of a navigation issue can be found on YouTube. Once videos on YouTube get a specified number of views, an account holder is paid. This transaction occurs based on agreement where, (1) the account must be an AdSense version, (2) the YouTube authority must review the channel, and (3) the channel must have the necessary permissions for advertising.

This information is included in the help section of the YouTube interface; see Figure 1. This figure indicates that for a YouTube user, they must locate this setting in the help section and proceed to navigate through to set up the account. It is argued that this process in non-trivial for average users and that this type of example is numerous on the web. This paper documents a novel real-time system where the steps in setting up such an account for instance, will be automatically suggested in a supplemental navigation bar. This improves a user's experience and improves on the functionality of the application.

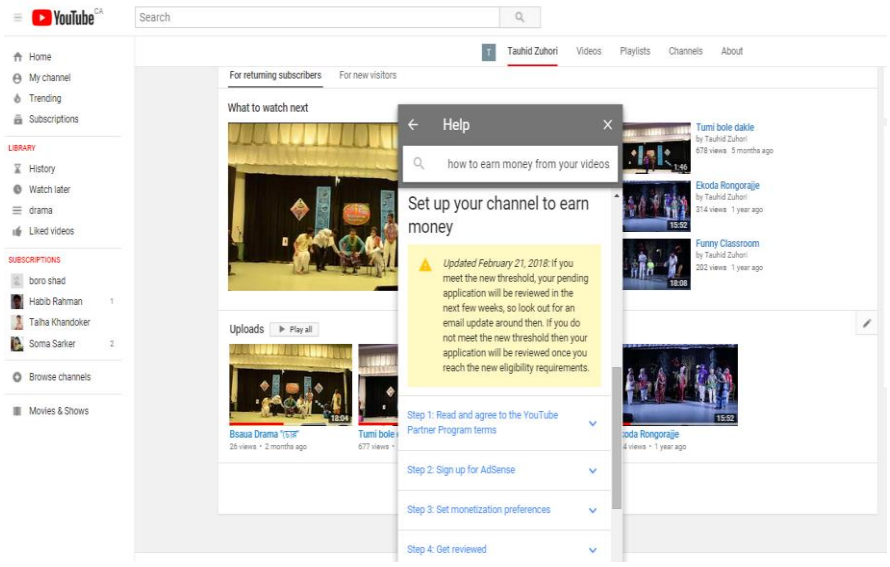


Figure 1. Screenshot of "YouTube" Account Settings

Our proposed assistance system can save browsing time for web users. For instance, visa application processes are a lengthy process. Recently part of the process has been digitized. A perfect example is the 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 2 illustrates that. The website combs through the answers and provides a result that either the user is eligible or not for a visa. This is followed by a process of uploading a number of 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 3 illustrates this with a screenshot. However, this tedious and cumbersome process can be greatly improved with the proposed system. The system will direct the new user based on different recorded (previous) user experiences. This improves user experience by being more accessible and efficient.

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 users requested link as input and providing appropriate suggestions.

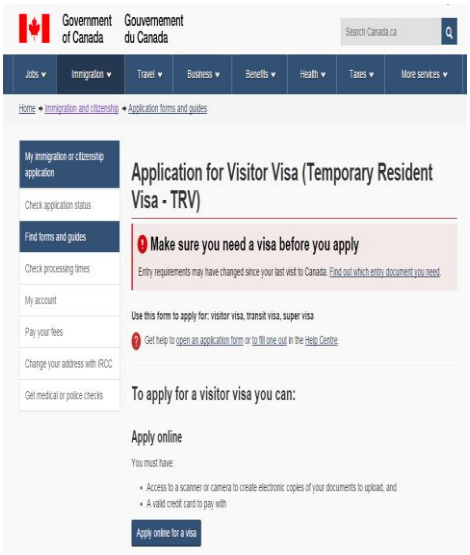


Figure 2. Screenshot of "CIC- Apply online" page

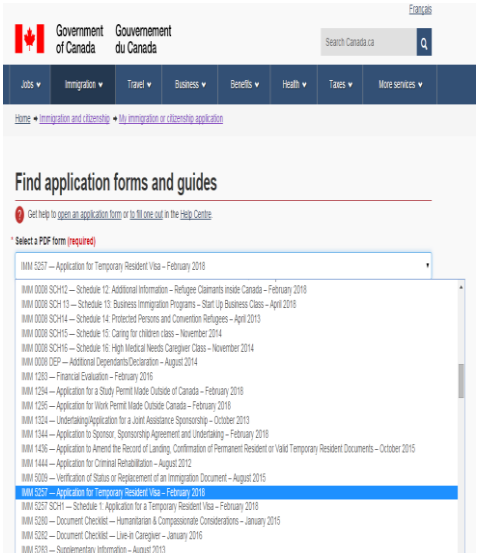


Figure 3. Screenshot of "CIC form selection" page

Our paper makes the following major contributions:

- We build a real-time suggestion generation system for websites which 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 web site before opening and exploring.
- 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, the "University of Alberta" websites, 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.

The rest of the paper is organized as follows.

In section 2, we discuss briefly previous research that is related to our work. Then in section 3, we discuss the methodology of our proposed approach. After that, in section 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.

2. RELATED WORK

Much work has been done on Web Usage Mining. In general, three major orientations can be found in this research area: analyzing user behaviors, clustering of the users of a website and web link prediction and recommendation. We present 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 group similar web pages by the distribution of time spent browsing a web. This distribution o is different for dissimilar types of page such as registration form, index pages, news, description of products etc. 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 user 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 users 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 reposting rate is much higher than the posting rate in the blogging site and males are more actively involved than females. 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 pretend web application www.findyourhouse.com.

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 web site 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. Wan et al. (2010) transfer the clustering task into a Chaotic optimization problem by proposing a CAS based clustering algorithm They compare their proposed algorithm with k-means clustering algorithm in terms of average intra-cluster distance and average inter-cluster distance. They also claim that their system has three advantages over k-means clustering: 1) finding a global optimum clustering results, 2) No centroid is needed for selecting the initial steps, and 3) good algorithm stability. Gang et al. (2016) bring a new era in this type of research by making clusters visible. They identify the clusters of similar users by partitioning a similarity graph where the 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 hierarchy within user clusters and produces features for visualizing and understanding captured user behaviors. For evaluating their system, they present two case studies on two large scale clickstream traces from real social networks. They can identify dormant users and hostile chatters with their system.

In the third area of web link prediction and recommendation, Shahriary et al. (2015) propose a ranking algorithm for detecting communities in signed graphs. They test their algorithm on three large scale datasets; Epinions, Slashdot and Wikipedia. Liu et al. (2007) propose an approach to classify user navigation patterns and predicting the user's future request. Their approach is based on combined mining of web server logs and the content of the retrieved web pages. They capture the textual content of the web page by extracting the character of N-grams. Then they combine it with web server log files to derive the user's 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 a number of cluster and then applying a collaborative filtering algorithm. Then they use the similarity between the clusters based on the links between them. They experiment on a number of real datasets and show that their proposed method is better than the previous methods. 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. The detailed record in NFP data includes a user's ID that can uniquely identify a user, accurate time (in hour or minute), the App's Id that the user visited in the time, and the page view number that represents the user visited frequency in the time, the App's category Id etc. They construct a User-App bipartite network, and transform the App-usage prediction to a link prediction problem in the complex network which can focus on extracting missing information. For testing, they collect 4-days of NFP data from ISP's Operational network. Gurini et al. (2015) 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 in Social media. They define a sentiment-volume-objectivity function and their method utilizes on three concepts;
1) The construction of the graph, one for each topic cited by the user, 2) the detection of SVO-based latent communities through clustering technique, s and 3) the computation of the global similarity between users of the networks. They use real-world datasets for their experiment. Adeniyi et al. (2016) present a study of automatic web usage data mining and recommendation system based on current user behavior through their click stream data. They use a K-Nearest-Neighbor (KNN) classification method for training the model. They identify the client's/ visitor's clickstream data at first. Then match it to a particular user group at a particular time. To achieve this, they extract the RSS address file, clean and format the file, and finally group the file into a meaningful session. Wang et al. (2008) develop an online navigation aid using collaborative recommendations based on graph theory. They utilize the past data of the user for that and use the server log file as input.

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 huge immediate work load. 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 necessary required in making suggestions. This has been solved in the new system where the real-time suggestions are done based on behavioral models that are update on real-time. Besides this the recommendations of previous studies are conducted offline and can therefore accessed by the web site developer only. Due to the effective development of the system, a user will be providing with accurate suggestions even for different browsers and different platforms. 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. OUR PROPOSED APPROACH

We start this section by discussing the model generation process. We use the user's navigation path as input and then generate a model using a Discrete Time Markov Chain (DTMC) inference process that represents the user behavior of a website. Then using this model, we predict the next visited (hyperlinks for the users of the website. Finally, we build an automated system that can track the user's current requested web page and according to that produce suggestions by using the DTMC of the website.

3.1 Model Generation

While interacting with web applications, users behave in different ways. The sites that they visit and their browsing history in general is informed by specific needs at a specific time. This brings up the question, how we optimize the users interface or browsing capabilities by using more advanced technology. The simple answer to this is the model representation of specific user behavior and latent patterns by use of a graphical and traceable representation. A common suggestion to this procedure is tracking user browsing history and using this to generate models that contain different paths. The Bear, Ghezzi et al. [2014], framework generates models of user behavior for user-intensive web applications. It uses a set of probabilistic Markov models from users' browsing history. It utilizes two classifies that are used to classify the user based on two values, the user-agents (e.g. Mozilla, Firefox) and secondly the user's location (this can be extracted by geolocation). In order to improve the services more classifiers can be used to build a single model entity. The importance of classifiers in this process is to extract domain specific information about the users. In order to manage this daunting task, the framework's engine incrementally generates a set of Discrete Time Markov Chains (DTMCs) representing specific user's behavioral models. DMTCs in this case are probabilistic finite state automata which utilize a Markovian process. While this is in progress the framework registers the DTMCs with a numeric value called a reward. The reward register for a single entity indicates the quantitative value (benefit) of visiting a specific web page or a specific state of a described model. In this particular framework, rewards are **manually** determined and assigned by the system designer to the states of the models. Hence, this requires the construction of literally billions of reward values to be constructed **manually** for a large, on-line web service per day! This makes the approach impractical. So, we use the automated reward calculation procedure proposed by Emam et al. (2018). However, the basic concept of using a DMTC model seems sound. This means that a DTMC with a reward is a tuple (S, P, L, ρ) where: i) is a set of states, and $s_0 \in S$ indicates the initial state; ii) $P: S \times S \rightarrow [0,1]$ is the probabilistic matrix indicating the probability of the occurrence of a transition between two connected states. iii) L is a function which maps a state to a set of atomic propositions (The unique links of a website). And iv) ρ is a reward function which associates a non-negative number with each state.

The framework analysis engine then reports the results of the reward by evaluating the properties of the interaction patterns against the inferred models. In order to **automate** this process, we use reinforcement learning. This is generally based on performing an action (i.e. mapping situations to actions). During each (real-time) iteration, agents learn by observing the current environment, inferring the environment's state and executing an action. The result then guides the agent to the next state. As a result, the system makes a transition to a new state as informed by the agent. This process is repeated, constantly maintaining the state of the system

as part of Markov Decision Process (MDP). The MDPs therefore can be categorized as stochastic extensions of finite automata or Markovian processes which are detailed by actions and rewards characterized with actions, transitions and states. In order to understand this process better the following definitions have been formulated.

3.1.1 Markovian System

A system can be defined as Markovian if the execution of an action does not depend on previous actions and visited states (i.e. it depends only on the current state and status). An MDP contains: 1) A finite set of states $S = \{s_1, s_2, \dots, s_N\}$, where N is the number of states; 2) A finite set of actions $A = \{a_1, a_2, \dots, a_k\}$, where k is the size of the action space; and 3) The transition function $X: S \times A \times S \rightarrow [0,1]$ which computes the probability of reaching the state s' by performing action a in state s and is denoted as $(X(s, a, s'))$. So s_t denotes the state at time t. Therefore, the definition translates to the following formula;

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

3.1.2 Reward Function

R specifies the reward the agent receives by performing an action. So, $R: S \times A \times S \rightarrow \mathbb{R}$ presents the reward function that computes the immediate utility of an action to define the model of the MDP.

3.1.3 Policy

This is the function that determines which action can be taken in each state to maximize the reward function. In other words, the stochastic policy $\pi: S \times A \rightarrow [0,1]$ is a mapping from each state s and action a to the probability $\pi(s, a)$ by performing an action a when in state s.

$$\pi(s, a) = P(a_t|s_t = s) \quad (2)$$

3.1.4 State-Value Function

The Value Function$V^{\pi}(s)$, specifies "how good" it is for the agent to be in a given state. We can define the value of a state under a policy π, formally $V^{\pi}(s)$, as:

$$V^{\pi}(s) = E_{\pi}\{R_t|s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s\right\} \quad (3)$$

For estimating the value function, we use Q-Learning, a method that is used to estimate Q-value functions in a model-free fashion. As observed above, we require a detailed transition and rewards model for successful application of the technology. There is a need therefore of sampling and exploring to learn the required model for this case. In order to achieve this, Q-learning estimates the agent's Q-value function based on an action's Q-value estimate. This is basically an incremental process as detailed by the below formula,

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

Where, α $(0 < \alpha \leq 1)$ is the learning rate which determines the extent to which new information can override old information.

We have used Q-learning to estimate the reward value in this work as the Q-value function can converge an optimal policy and also estimate the state-action value function in free model problems. The Q-value function is an appropriate option here for two reasons; 1) It can incrementally learn the values from the current state of the model; and 2) It can easily be configured to generate meaningful reward values for web applications.

The equation (5) shows how the Q-value function has been used in our proposed approach to calculate the value of the reward at state, s_i, which is called $\rho(S_i)$.

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

3.2 Generating Suggestions

Prism (Henquies et al. ,2012), a probabilistic model checker, is utilized in the proposed system for generating suggestions for users. This is enabled by the use of a real-valued query: $\mathfrak{R}$ query [rewardprop] and the reachability reward F, as it considers the reward value of all the states up to the chosen label. Suppose a user visits a link in our sample test cases. For instance, "Bear tracks" in the University of Alberta website. Then the user can get suggestions derived from the link "Bear tracks". This methodology is interpreted as the following property to provide accurate reward (suggestion).

$$\{\}\mathcal{R} =_? [F\ Bear\ Track] \quad (6)$$

The corresponding URLs of the output reward values are displayed as suggestions in the suggestion bar. In cases where the web pages lack an auto generated suggestion, the system will easily return "Home" page as the default suggestion.

3.3 Real-Time System Design

We design our real-time browsing assistance system using Algorithm 1. We take the user behavior matrix of the website that is produced by the discrete time Markov Chain inference process as input. A user can use our system as an extension within their web browser.

Algorithm 1. Real-time suggestions producing algorithm

Input: Requests submitted by the user to web server

Output: A number of suggestions

1. Set, p=0, $\rho(S_0) = 0, \rho(S_e) = 0$, AP=NULL, L={$S_0, S_e$}, t=0, RU(0)=0
2. **do** the following steps until no request is found?
3. Extract the IP, Timestamp (TS), Requested URL(RU), browser name (B) from the request. if RU is not found in the AP list then insert it. Set, S_t= RU(t) and S_{t+1}= RU(t+1)
4. **if** TS(RU(t+1)-RU(t))>=30||IP(RU(t+1)!=IP(RU(t))||RU(B)!=RU(B) **then**,
5. Set edges between S_0 to S_{t+1}
6. **else**
7. set edges between S_t to S_{t+1}
8. **end if**
9. **if** TS(RU(t+1)-RU(t))>=30 **then**,
10. Set edges between S_t to S_e
11. **end if**
12. Set $P_{(Source,destination)} = \frac{transition\ between\ S_{source}\ to\ S_{destination}}{Total\ number\ of\ transition\ of\ State\ S_{Source}}$
13. RU(t+1)=chrome.tabs.query('active':true,'lastFocusedWindow':true);
14. document.write(SuggestionList); / using section 3.2
15. Set, a_{t+1}= RU(t+1)
16. Update the reward value, $\rho(S_{t+1})$ using Q-learning
17. t=t+1;

4. EVALUATION

In this section, we include an evaluation and validation of our system. We start the section with a brief description about the datasets used in this endeavor. Then we represent a user study. After that we present case studies to demonstrate how users of the website benefit 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 a cross validation of the results.

4.1 Clickstream Dataset

We utilize 2 datasets of server log files from the University of Alberta website. We do not use any personal information of the users for ethical anonymity. Figure 4 is used to visualize the 2 types of datasets.

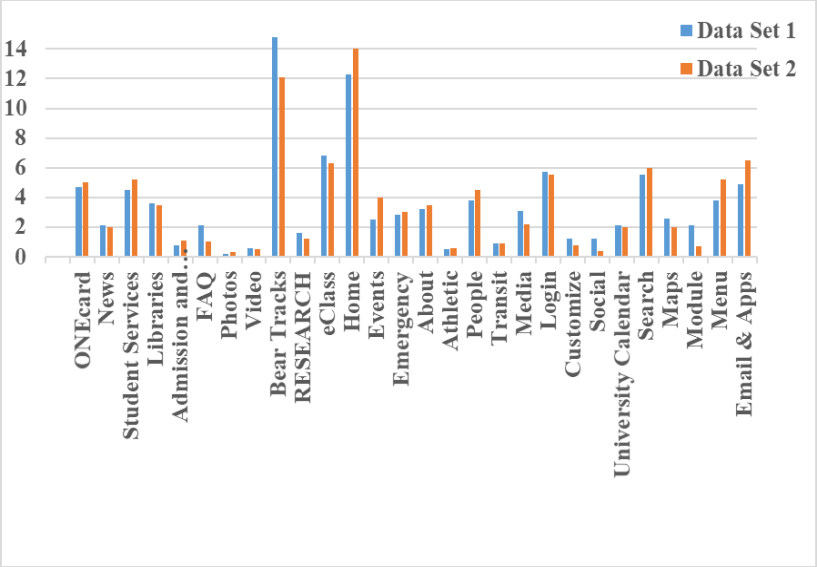


Figure 4. Dataset Visualization

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.

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. The system decreases 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 the questions 1 to 5, where 1 means strongly disagree, and 5 means strongly agree. We collected data for these questions after 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). Table 1 summarizes the findings on the factors for "UofA" (40 users).

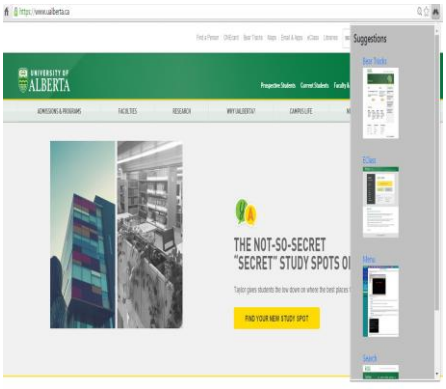
Table 1. Test analysis("UofA") for Factors Data conventional system and our proposed system

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

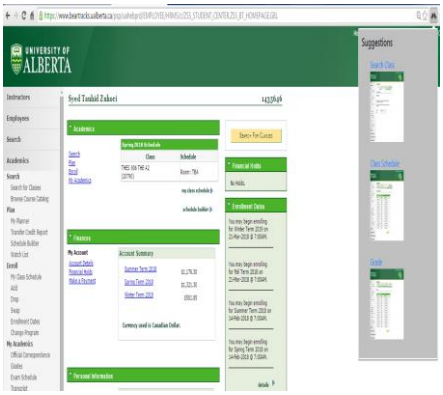
From the test analysis, we observe that there is a statistically significant difference in user experience between the conventional system and our proposed system. The result of GFD, U-test and Cliff's Delta indicates that our proposed system can provide very helpful suggestion that can improve the navigation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

4.3 Case Study

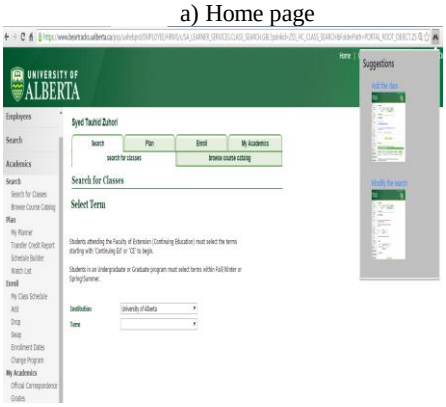
Next, we present an in-depth analysis on users of the website, "University of Alberta". Due to lack of space, we focus on three types of users of the web site. In these case studies, we show how the user can benefit from our real-time system.



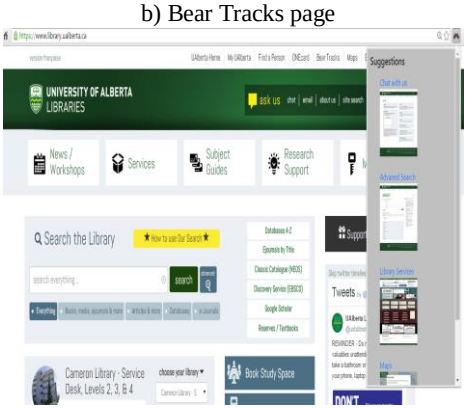
a) Home page



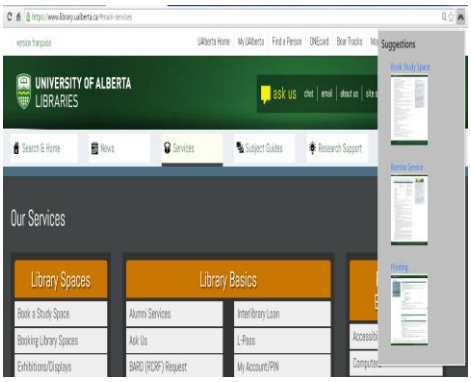
b) Bear Tracks page



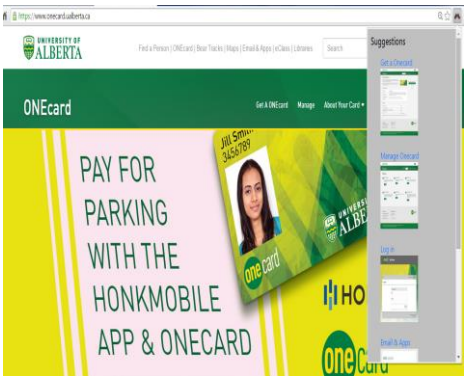
c) Search course page



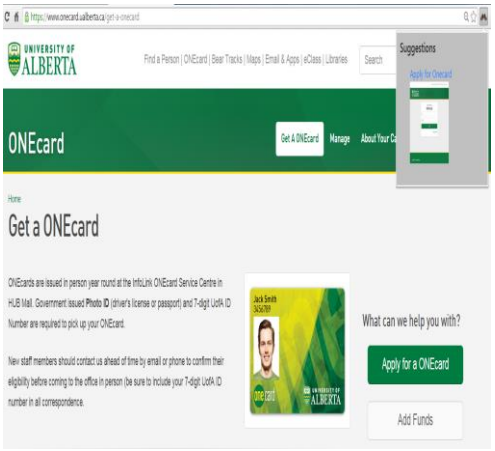
d) Libraries page



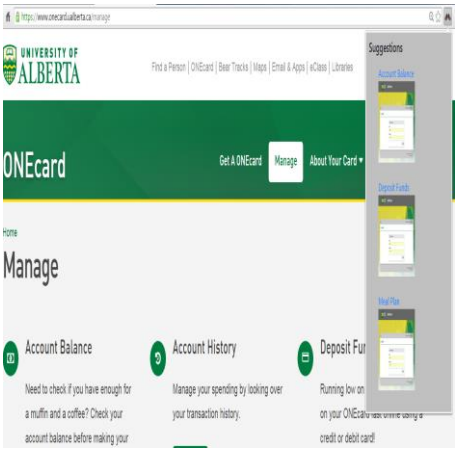
e) Library services page



f) ONE card page



g) Get a one card page



h) manage one card page

Figure 5. Screen shot of our automated system suggestions for University of Alberta website

Case Study 1 ("Bear Track" users of University of Alberta website): "Bear Tracks" is one of the important links in the Website. Students use this link to register or drop a course, see their grades etc. From figure 5(a) we see that our system suggests the "Bear Track" link as it is one of the most visited links and it shows that a login is needed to visit that page. After the login three suggestions are displayed, (1) a user can search a class for registration, (2) check the class schedule or (3) check their grades (figure 5(b)). If a user selects the "Search class" option, then there are two suggestions: (1) modify the search or (2) add that class to their class list, figure 5(c).

Case Study 2 ("Library" users of University of Alberta website): There are many useful links at the Library page including an online chat system. Figure 5(d) shows our automated system suggesting a library user to chat with a representative, the link to the advanced search option on the library database and other library services. If users go to the library services option, then according to figure 5(e) they can see the most popular service links provided by the University of Alberta.

Case Study 3 ("One card" users of University of Alberta website): "One card" is the ID card at the University of Alberta. From figure 5(f) we can see that users are suggested to go to the link to get a one card or manage a one card. Users can visit the main links "Email & Apps", "Student Services" etc. If a user wants to get a one card, then they are suggested to apply for it, figure 5(g). On the other hand, if a user already has a one card then they can choose the "Manage One card" option. Figure 3(h) shows that there are three suggestions at that option: (1) check account balance, (2) manage meal plans, and (3) deposit funds at "One card".

4.4 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.

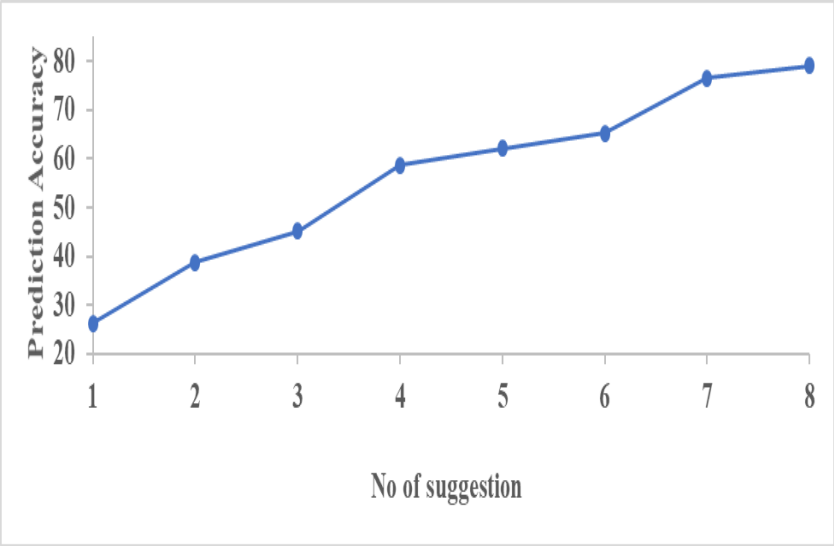


Figure 6. No of Suggestion vs Prediction Accuracy (UofA)

Figure 6 represents the prediction accuracy with respect to Number of suggestions for the University of Alberta website. As indicated in figure 6, with the maximal number of suggestions increases from 1 to 8, the accuracy also increase. 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.

Figure 7 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, out of the 28 pages that are used for the test case, the most predicated based on accuracy, is the "Home" page with a 92%, and the "Login" page with 86%. This makes sense since most users visit those pages to enable navigation to other webpages.

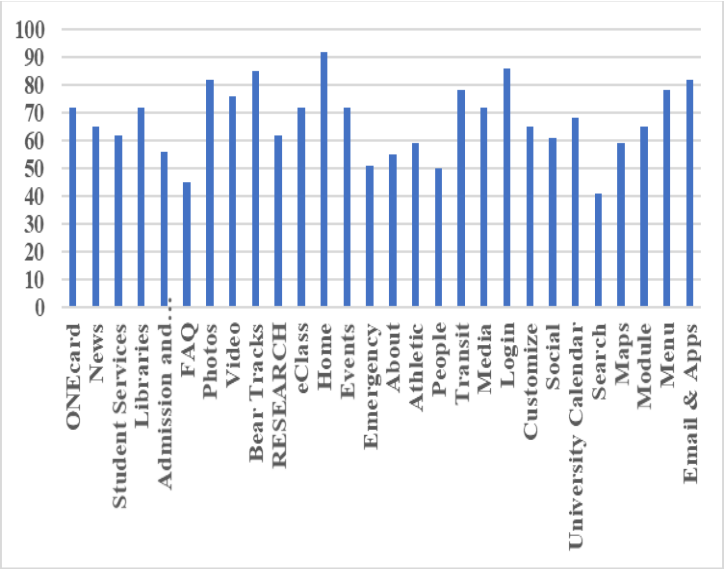


Figure 7. Unique links vs Prediction Accuracy (UofA)

After that, we test how well our proposed system can 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 8 represents the prediction accuracy with respect to the percentage of users for the "UofA" dataset.

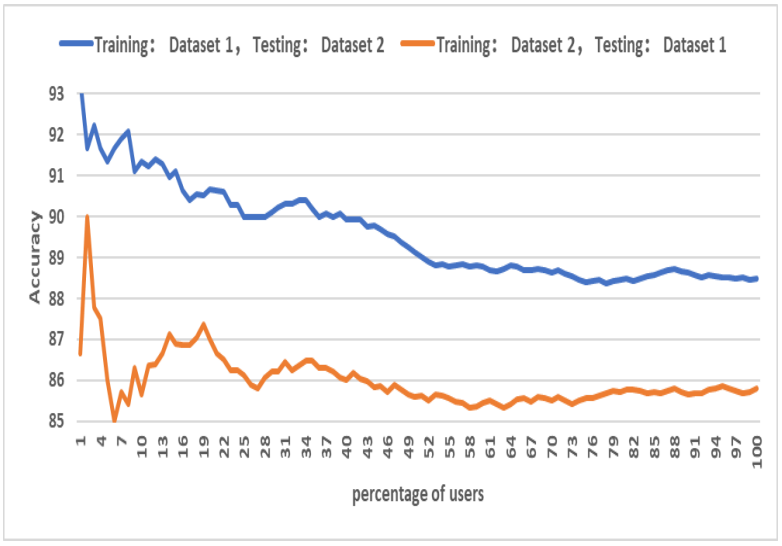


Figure 8. Accuracy in "UofA"

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 predication. This is then repeated for 2%, 3% incrementally to 100%. Except 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 first case, dataset 1 is considered as training and dataset 2 as testing. We consider the opposite in case 2. From the figure 8 we observe that in case 1, the system stabilizes at around 80.69% and the accuracy lies between 74% to 81%. For case 2, the accuracy lies between 75% to 82%.

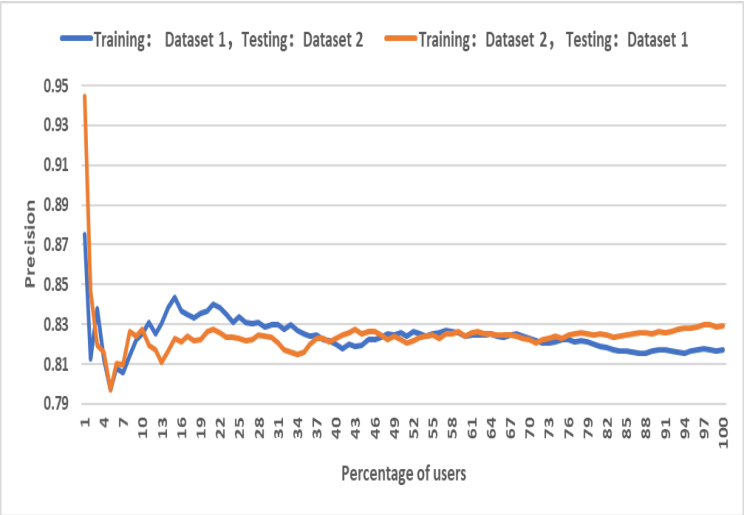


Figure 9. Precision in "UofA"

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 in fact five, then the recall of the system would be 4/5 or 0.8. Figure 9 represents the precision of "UofA" datasets where the average precision for case 1 is 0.817 and case 2 is 0.829.

In the case of recall, figure 10 represents that at "UofA" dataset the average recall is 0.854 for case 1 and 0.865 for case 2.

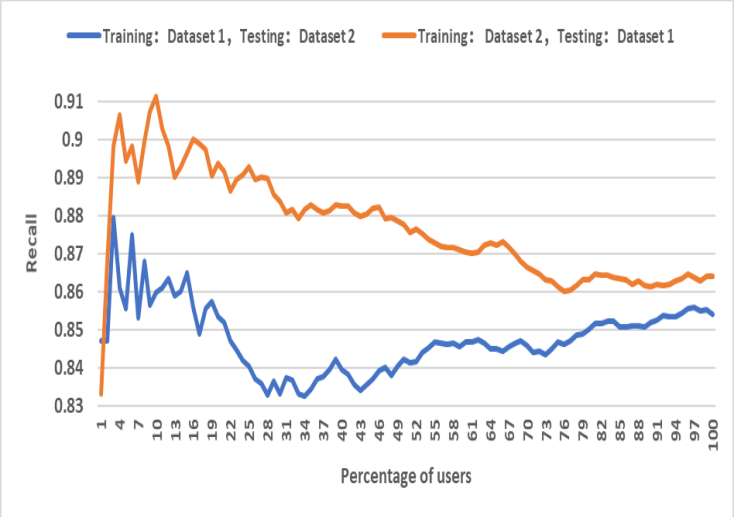


Figure 10. Recall in "UofA"

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

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

"UofA" system					
Accuracy		Precision		Recall	
Case 1	Case 2	Case 1	Case 2	Case 1	Case 2
0.7937		0.6252		0.1587	

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 2 shows the value of 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 same as another case with significance level 5%. So, from this experiment we can decide that, if we exchange our dataset at training and testing period then the system performance will remain "similar".

5. CONCLUSION

This paper introduces a real-time system to assist the web-site users. To achieve this, our proposed system captures the requests submitted by web users to the web server in real-time and generates an inference behavioral model using the Discrete Time Markov Chain inference process. After that, it produces navigation suggestions to the users and updates the inference model by using a Markovian Decision Process. For evaluating our system, we conduct a user study, case studies on different types of users 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. Beside this, the performance indicators represent good 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 better information service. Using our proposed system, the website developer can also improve the websites hyperlink structure by better understanding and predicting users' navigational behavior.

REFERENCES

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.

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).

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).

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

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).

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).

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

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).

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).

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).

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).

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

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

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

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

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

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).
Wang, Y., Dai, W., Yuan, Y.: Website browsing aid: A navigation graph-based recommendation system. Decision Support System, 45, pp 387-400, Elsevier (2008).
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).
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).
Wan, M., Li, L., Xiao, J.: CAS based clustering algorithm for web users. Nonlinear Dyn.: 61, 347-361 (2010).
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).
Zhu, J., Hong, J., Hughes, G.: Using Markov Chain for Link Prediction in Adaptive Web sites. Soft-Ware, pp 60-73, Springer (2002).
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.
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.
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.
C. Szepesvári, Algorithms for Reinforcement Learning, vol. 4, no. 1. Morgan & Claypool Publishers, 2010.
M. Wiering and M. van Otterlo, Reinforcement Learning (State of the Art). Springer, 2012
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.
L. C. Stuart, "User Modeling via Machine Learning and Rule- Based Reasoning to Understand and Predict Errors in Survey Systems," 2013.
M. Virvou, C. Troussas, and E. Alepis, "Machine learning for user modeling in a multilingual learning system," pp. 292–297, 2012.
D. Henriques, P. Zuliani, and E. M. Clarke, "Statistical Model Checking for Markov Decision Processes," vol. 1041377, no. 1041377, 2012.
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.