Simulating User Interaction and Search Behaviour in Digital Libraries

Saber Zerhoudi[1], Michael Granitzer[1], Christin Seifert[2] and Joerg Schloetterer[2]

[1]*University of Passau, Passau, Germany*
[2]*University of Duisburg-Essen, Duisburg, Germany*

Abstract

Providing a satisfying user experience is key to successful digital libraries. An efficient search user interface design requires a detailed understanding of user behaviour to match their needs. A common approach to modelling users is to transform the recorded users' navigation behaviour to Markov models and analyse them to provide personalised content. However, personalisation based on log data is hard to achieve due to the lack of sufficient daily user interactions in comparison to web search engines, even when achieved, it's usually on the cost of users' privacy and the confidentiality of their personal information potentially leading to privacy violation. In order to allow data analysis while retaining users' privacy, we propose a Markov approach to simulate user sessions and help reducing the amount of collected user data while preserving the profiling efficiency. Specifically, given log data of search sequences performed by users in a digital library, we explore basic, conditional, contextual, time-aware and query-change Markov models to simulate similar search sequences. Our results on an academic search engine log dataset show that our methods reliably simulate global and type specific search behaviour. Simulated search sequences representing an approximation of real behaviour can be used to generate log data at any desired volume and variety such that A/B-testing digital libraries becomes scalable.

Keywords

Simulating user session, Markov models, User search behaviour, User modelling

1. Introduction

The main aim for search engines is to offer a satisfying user experience. Their success can be measured by how well they are able to predict user actions and provide users with the right content at the right time. Modelling user behaviour accurately also allows search engines to adapt their user interface design to match users' needs. This includes deciding where each user interface component should be placed and which content should be provided to the user. However, problems arise when improving platforms such as digital libraries (e.g., academic search engines) that aim to provide personalised content to their user-base, namely researchers and students. In fact, it is often not possible to build personalised user models due to the lack of sufficient daily user interactions in comparison to web search engines [1], and even when

IRCDL 2022: 18th Italian Research Conference on Digital Libraries, February 24–25, 2022, Padova, Italy

saber.zerhoudi@uni-passau.de (S. Zerhoudi); michael.granitzer@uni-passau.de (M. Granitzer); christin.seifert@uni-due.de (C. Seifert); joerg.schloetterer@uni-due.de (J. Schloetterer)
0000-0003-2259-0462 (S. Zerhoudi); 0000-0003-3566-5507 (M. Granitzer); 0000-0002-6776-3868 (C. Seifert); 0000-0002-3678-0390 (J. Schloetterer)

CEUR Workshop Proceedings (CEUR-WS.org)

available, such models would be generated at a significantly smaller scale and would be less accurate.

Recently, user behaviour has been subject to controversy debates in information security and privacy related areas, especially with the large portion of collected user data in web search engines and digital libraries. Analysing this wealth of information without privacy-concerns allowed - web search engines on a larger scale and digital libraries on a minor one - to improve their search experience. However, concerns started to rise regarding the usage of collected sessions and the impact of storing personal, privacy-sensitive data without the user knowing.

To reduce the amount of collected user data while preserving the profiling efficiency and help domain-specific search engines, where the amount of user data is limited, to provide accurate information to users while protecting their privacy and the confidentiality of their personal information, we propose to use simulation. Simulation offers a way to overcome the lack of experimental real-world data, especially when the acquisition of such data is costly or challenging [2]. Simulating user search behaviour also allows information retrieval systems (IR) to conduct A/B-tests for the purpose of monitoring user interactions and parameterising the a-priori distribution of search types using different back-end configurations and user interface variants (e.g., changing facets placement in the user interface to more prominent place) to improve the retrieval functionality.

To better simulate users' search behaviour and model their information needs, search engines offer contextual information with data such as previous queries and click-actions, which is very useful for understanding user search behaviour and improving the retrieval system configurations involved in the search process. In fact, digital libraries should consider user's browsing context (e.g., which users are more likely to formulate more queries) and sequentiality (e.g., which search actions are performed knowing that the user has formulated its x-th query) to improve the precision of results. Users in the same group are most likely to behave in a similar manner. Therefore, it gets easier to recognise users' intentions and simulate their behaviour when exploiting the available contextual information.

In this paper, we propose the use of Markov models to simulate global and search-type specific behaviour to support users in their search. Markov models have been widely used for discovering meaningful patterns in browsing data due to their good interpretability [3, 4, 5, 6]. In particular, they capture sequences in search patterns using transitional probabilities between states and translate user sessions into Markov processes.

Our contributions are the following: (1) we conduct experiments on a real-wold dataset and simulate user search behaviour using a basic Markov approach which we consider as baseline, (2) we then demonstrate that while using user's browsing sequentiality, the Markov model simulates user sessions marginally better than the baseline, (3) we also showcase that Markov model perform well when considering user's actions or the user's intent-level query-change as context, even when the contexts are derived based on common sense assumptions and (4) modelling the dwell time and using it as an additional feature for session simulation yielded better results. Our contributions also include (5) an empirical validation that utilises the Kolmogorov-Smirnov statistical test to compare the similarity between real log and simulated data distribution and (6) an evaluation technique that uses classification to assess the quality of simulated search session and calculate the model's improvement in comparison to the baseline.

2. Related Work

While Markov models can be used in other domains such as credit card fraud detection [7], computational biology [8] or cyber-security [9], we restrict the discussion here to understanding, modelling and predicting users' search behaviour on search engines.

Several Markov models variations and extensions were proposed for modelling user navigation behaviour. For example, Cadez et al. [10] used first-order Markov models to cluster user behaviour and reported some interesting insights (e.g., order in which users request pages) that improved the website which the data was taken from. Liu et al. [11] explored Continuous-Time Markov Processes for modelling users' browsing behaviour in web search to help computing page importance. Azzopardi introduced Search Economic Theory (SET) [12] to model the search process and estimate users' effort during a search process using a cost function. He compared the interaction costs of different search strategies using simulated user interactions. Hybrid-order tree-like Markov models [13], Selective Markov models [14] and variable order Markov model [15] were also used to extract users' interactions from logs and capture both small and large order Markov dependencies.

Besides, the most commonly used techniques are Hidden Markov Models (HMM) based. Hidden Markov Models are used to model exploratory search sessions. Lucas found HMM to be suitable for pattern recognition problems on the basis of sequential data [16] (e.g., credit card fraud detection, handwriting recognition or biological sequence analysis). However, working with HMM usually requires an understanding of the domain since we do not have control over the state transitions and the states are not completely observable. Kiseleva et al. [17] suggested to use a hierarchical clustering approach to decompose users' web sessions into non-overlapping temporal segments. Authors also identified temporal context and utilised it to predict future user's actions more accurately using Markov models.

Our discussion of related work reveals a strong focus on predicting and modelling user's search behaviour. Despite this fact, the application of Markov model as a simulation model outside the healthcare domain [18] is still in its infancy.

3. Methodology

The main purpose of this study is to simulate user search behaviour based on user's behaviour patterns. In other words, given log data of search sequences performed by users in a digital library, our goal is to simulate similar search sequences. In this work, we use Markov model as a simulation model. In general, the input for this problem is a sequence of search sessions produced by real users, and the goal is to build Markov models using different approaches that model user's search behaviour.

3.1. Basic Markov model

We propose investigating the use of Markov Chains to model the search dynamics. The theoretical model is based on first-order Markov models [19]. Our model is defined as a weighted directed graph, where user actions are represented as nodes and the transition probability between actions as edges. In fact, let $G = (A, E, w)$ where:

- A is the set of all possible user search actions including START/END node, with each action defined as state S_a,
- $E \subseteq A \times A$ is the set of all possible transitions between two actions,
- $w : E \to [0,1]$ is a function that assigns a weight w to every pair of states (S_i, S_j) which represents the likelihood of a transition from state i to state j.

Let X_k be the random variable that models actions in a user search session. Basic Markov approach models the transition probability between a pair of states using maximum likelihood estimation as follows:

$$P(X_k = S_b | X_{k-1} = S_a) = \frac{N_{S_a,S_b}}{N_{S_a}} \tag{1}$$

where N_{S_a} is the total amount of how many times the state S_a occurred in the training data and N_{S_a,S_b} is the amount of how many times the transition from state a to state b has been observed.

We use first-order Markov model as a baseline to model the global user search behaviour and simulate user search sessions in such a way that each action that is performed by a user corresponds to a state in the model.

3.2. Conditional Markov model

In conditional Markov approach, we aim to model the probability to perform an action given the current action and query, and the probability to formulate a query given the current action and query. We extend the work that has been done in [20] by adjusting it to the simulation scenario. We utilise the user search sessions in the log data to calculate the transition probabilities between different states resulting in a Markov model that takes into consideration the following definitions:

- $i \in \mathbb{N}$ user query: the i-th query formulated in a user search session
- $j \in \mathbb{N}$ user action: the j-th action performed for the i-th query in a user search session

Every user action in a search session is related to the preceded query formulation. With the exception of the first query formulation that marks the start of a user search session, every query is also related to the preceded action performed by the user.

Let X_k be the random variable that models query formulation in a user search session and Y_k the random variable that models user actions performed for the last query in a search session after k transitions. In order to model user sessions, we define the pair $(X_k = i, Y_k = j)$ in a way that it represents when a user performs j-th action after formulating an i-th query. We introduce two different types of transitions. The first one models the transition probability of formulating i-th query given that j actions were performed in the $(i-1)$-th query [20]:

$$P(X_k = i | X_{k-1} = i-1, Y_{k-1} = j) = \frac{P(X_k = i, Y_k = j)}{P(X_{k-1} = i-1, Y_{k-1} = j)} \tag{2}$$

and the second one models the transition probability of performing j actions given that i-th query was formulated [20]:

$$P(Y_k = j | X_{k-1} = i, Y_{k-1} = j-1) = \frac{P(X_k = i, Y_k = j)}{P(X_{k-1} = i, Y_{k-1} = j-1)} \tag{3}$$

where the first event in a user search session is a query formulation followed by either a user action or a new query reformulation. After k events, the transition probability is calculated using the Markov transition matrix M^k of k-th order.

In the basic approach, the random variable X_k represents the action/state (regardless whether it's a query or an action) and the transition probability from state a to b is $P(X_k = b|X_{k-1} = a)$. The Markov property (i.e., future state depends only on the current one) is also preserved. For the conditional approach, we add a second condition, i.e., the future action always depends on the current action and query, which makes it necessary to distinguish queries X_k and actions Y_k. We then have $P(X_k = i|X_{k-1} = i-1, Y_{k-1} = j)$ and $P(Y_k = j|X_{k-1} = i, Y_{k-1} = j-1)$ as transition probability.

We compute the transition probabilities based on these two definitions (i.e., Eq. 2 and 3) by looking at the current action and the current formulated query when trying to model the probability to perform another action or formulate another query.

3.3. Contextual Markov model

During a search session, a user performs different search actions to find documents that fulfill his/her information need. The technique that we propose here aims to categorise users into different groups based on their search behaviour. Search tasks are commonly divided into two major types of user's behaviour [21, 22]: (1) "exploratory search" where users tend to explore the search result list exhaustively, rephrase queries or extensively use filtering operations on the results, and (2) "known-item" where users only investigate the first few results and rephrase their queries quickly to match their need.

In this approach, we utilise search-type context that splits the training data into smaller portions. More precisely, we build a first-order Markov model for each search type (i.e., exploratory search and known-item) by adopting context and compare the simulation's performance to the Markov model built from the whole data. This would allow us to evaluate the impact of context on the accuracy of simulated sessions.

Kumaripaba et al. [22] extended the work of Marchionini [21] and provided a few simple indicators of information search behaviours to categorise users into exploratory and known-item searchers. They showed that the most distinctive indicators that characterise exploratory search behaviours are query length, maximum scroll depth, and task completion time. They proved empirically that exploratory tasks can be distinguished within in the first query session by these indicators. Following their findings, we identify known-item search sessions by fine-tuning their indicators to fit our dataset and capping the first query duration to 200 seconds, the dwell duration of the first three actions to 120 seconds, the cumulative actions tally in the first query iteration to 4 actions and the the task completion time to 800 seconds. The remaining user sessions are considered as exploratory search.

The transition probability between any two states S_a and S_b is modelled similarly to (Eq. 1).

3.4. Time-aware Markov model

Our time-aware Markov model addresses time distribution between actions by taking the dwell time spent between each transition into account. We assume the existence of a distinctive distribution that governs how much time a user needs to move from state a to state b for each possible transition $S_a \rightarrow S_b$. Each transition time is collected from training data and used to estimate the time distribution of that transition. In order to calculate this time distribution, we utilise the gamma distribution which is governed by two parameters [23, 24].

Let Y be the random variable that models time between transitions. Y is gamma-distributed where θ is the scale parameter and k the shape parameter. Y is denoted by: $Gamma(k, \theta) = Y \sim \Gamma(k, \theta)$. The probability density function of gamma distribution can be expressed as follows [25]: $f(x; k; \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-x/\theta}, \quad x > 0; k, \theta > 0.$

Let N be a set of independent observations $(x_1, ..., x_N)$ of transition times between two states (S_a, S_b). The likelihood function is:

$$L(k; \theta) = \prod_{i=1}^{N} f(x; k; \theta) \tag{4}$$

and thus, the maximum likelihood function with respect to θ is *(obtained by setting the derivative of the log of equation* (4) *to zero)* $\hat{\theta} = \frac{1}{kN} \sum_{i=1}^{N} x_i$, and similarly the maximum with respect to k, $\ln(k) - \psi(k) \approx \frac{1}{2k}\left(1 + \frac{1}{6k+1}\right)$ which can be estimated from the training data. With the gamma distribution parameters estimated to fit the data, we aim to represent the dwell times in a realistic manner.

Similarly to the Basic Markov, time-aware Markov approach models the transition probability between any two states S_a and S_b using maximum likelihood estimation as follows: $P(X_k = S_b | X_{k-1} = S_a) = \frac{N_{S_a,S_b}}{N_{S_a}}$, where N_{S_a,S_b} is the number of times we observed a transition from state S_a to S_b and N_{S_a} is the total number of transitions from state S_a to any other state in the training data.

In this experiment, we seek to describe user action sequences while taking dwell time into consideration. We use the maximum likelihood to estimate the parameters of the gamma distribution for every transition from the log data. We then used the most likely transition time as a feature.

3.5. Query-change Markov model

Users tend to generate search sessions with borderline behaviours and mixed characteristics. In fact, users may have precise search goals, yet the search process is not straightforward. Several studies have shown that the query change which occurs in a search session is heavily related to the contents of previously viewed search results [26, 27, 28]. Therefore, looking into how users change their query from q_{i-1} to q_i can partially explain how they express their dynamic information need in the session. As consequence, such terms may be used for categorising users into finer groups to better simulate search-type specific sessions.

In this approach, we expand the findings of Huang et al. [29] and adopt user's session-level query change process as an implicit indicator for categorising search behaviour and estimating the change in user's information need. We develop a syntactic taxonomy by analysing the query change Δq_i as the syntactic change which occurs between two consecutive queries q_{i-1} and q_i: $\Delta q_i = q_i - q_{i-1}$.

Let $W(q)$ be the bag of words for q and q_i, q_{i-1} two successive queries. A query q_i can be decomposed into three main parts in relation to q_{i-1}, namely *theme*, *added* and *removed terms* and written as follows: $q_i = \Delta q_i^{=} + \Delta q_i^{+} - \Delta q_i^{-}$, where: $\Delta q_i^{=} = \{w \mid w \in W(q_i), w \in W(q_{i-1})\}; \Delta q_i^{+} = \{w \mid w \in W(q_i), w \notin W(q_{i-1})\}; \Delta q_i^{-} = \{w \mid w \notin W(q_i), w \in W(q_{i-1})\}$. We refer to the common terms that appear in both query q_i and q_{i-1} as *theme terms* and denote it ($\Delta q_i^{=}$) as they usually represent the main thematic of a session. *Added terms* (Δq_i^{+}) represent the new terms that users add to the previous query and *removed terms* (Δq_i^{-}) refer to the terms that users delete from the previous query as they seek to improve bad performing queries. *Theme terms* ($\Delta q_i^{=}$) are generated using the longest common subsequence (LCS) approach [30] in both queries q_i, q_{i-1}. The LCS can be the common part or parts of two queries as long as the subsequence appears in both queries in the same relative order but not necessarily continuous. For instance, $S_1 : q_1 \longrightarrow q_2$ is a search session where q_1 = "tire recycling technique in Europe" and q_2 = "tire recycling facilities in Europe". $\Delta q_i^{=}$ = "tire recycling in Europe", Δq_i^{+} = "facilities" and Δq_i^{-} = "technique".

In query-change Markov model, we employ queries as states. Specially, we define the pair $(X_k = i, Y_k = j)$ in a way that it represents when a user performs j action (i.e., keeping, adding or removing query terms) after formulating an i-th query. The transition probability of formulating i-th query given that j actions were performed in the $(i-1)$-th query is modelled as follows:

$$P(X_k = i | X_{k-1} = i-1, Y_{k-1} = j) = \frac{P(X_k = i, Y_k = j)}{P(X_{k-1} = i-1, Y_{k-1} = j)}. \tag{5}$$

Similarly to the first type of transition in the conditional approach, we compute the transition probabilities based on the first definition (Eq. 2) by looking at the current query-change (i.e., adding, removing or keeping query terms) and the current formulated query when trying to model the probability to formulate another query.

4. Experimental Setup

4.1. Dataset

We use Sowiport[1] *User Search Session data set (SUSS)* [31] for our experiments. The digital library provides records and information for the social sciences in English and German. We used data that was collected over a period of one year (from April 2014 to April 2015). The dataset contains over 480,000 individual search sessions and around 8 million log entries. we filter logs to remove any session that does not contain a query (i.e., users having searched nothing) or has invalid query annotations. In order to describe users' search actions, SUSS offers a list of 58 different actions that covers all user's activities while interacting with the interface of the digital library (e.g., formulating a query, clicking on a document, viewing the full document's content, selecting a facet, using search filters, etc.). For each user interaction, a session id, date

[1] http://www.sowiport.de

stamp, length of the action and other additional information are stored to describe user's path during the search process.

4.2. Evaluation Metrics

4.2.1. Kolmogorov-Smirnov-based Evaluation

Several statistical tests exist to evaluate the goodness of fit of two statistical models such as Likelihood ratio and the Person chi-square [32, 33, 34]. In this work, we utilise the Kolmogorov-Smirnov goodness-of-fit test [35] with its two-sample test *(KS-2)* variant. The advantage of this test is that it is one of the most useful and non-parametric methods of comparing two sample distributions. Essentially, we test the null hypothesis that the two independent samples belong to the same continuous distribution and proceed with calculating the absolute value of the distance between two data samples which we refer to as the test statistic d to compare their distribution for similarities. The test statistic d is calculated as follow:

$$d = \sup_x |E_{n1}(x) - E_{n2}(x)| \tag{6}$$

where, $n1$ and $n2$ are observations from the first and second sample respectively. $E_{n1}(x)$ and $E_{n2}(x)$ are the empirical distributions of the first and second sample, obtained from n1 and n2 observations. We then compare the test statistic value against the critical value derived from the KS-2 table value to either accept or reject the hypothesis. The null hypothesis is accepted when the test statistic value is less than the table value and rejected otherwise.

4.2.2. Classification-based Evaluation

In addition to the KS-2 test, we define a classification-based evaluation to (1) evaluate the simulation performance of our models and (2) calculate their improvement in comparison to the baseline.

In order to evaluate how good our model simulates user search behaviour, we first develop a set of features that represent the sequentiality of a user search session in the form of a feature vector. Then we train a classifier to distinguish simulated sessions from real log data sessions and report the results.

Inspired by the findings in [36] about what kinds of engineered features are best suited to various machine learning model types, we develop a set of features that represent the sequentiality of the search session (i.e., query search types; search types; advanced search types; clicking, viewing and exporting actions) and discard those that only describe the user's overall search behaviour (e.g., tally of search actions, queries formulation and clicks).

We used one-cold-encoding to indicate the presence of a feature (i.e., (0) if present and (1) if not) and ordered features in the sequence (i.e., i_feature where i refers to the sequence order of the query in a session , e.g., 1_search, 2_view_record). The resulting feature vector has a higher dimensionality (170) than the raw feature categories (58) due to the inclusion of the sequence order in the binary feature vector. Fig.1 gives an example of a user session in the form of a feature vector.

["1_search", "1_search_change_paging", "1_view_record", "2_search", "2_search_change_paging",
"3_search", "3_view_record", "3_view_description", "end"]

Figure 1: Example of a user search session in the form of a feature vector.

Another important feature that we also considered in our third approach as described in section 3.4 is the dwell time. Dwell time in a search action refers to the amount of time that the user spends in between the current and the future action (formulating a query, browsing a search result page, assessing a document, end of session.. etc.). We estimated the dwell times using gamma distribution for each state transition and added them to the list of features described above.

Each user session is converted to a feature vector, labelled and fed to a classifier. This task was repeated separately for each of the Markov approaches defined in section 3. We combined an equal amount of data from log sessions and simulated sessions for a balanced classification and split our sequential dataset of user's actions into a training set (consist of 80%), on which we trained the classifiers, and a test set (consist of 20%), on which we evaluated the model.

To evaluate how well the classification results generalise, we adopted 10-fold cross-validation that we performed in all classification experiments. We ran all experiments using the simulated sessions that we have generated from each approach and reported the average score over 10 independent runs. We used the five most popular algorithms in binary classification, namely, Logistic Regression [37], Support Vector Machine [38], K-Nearest Neighbors [39], Decision Trees [40], Random Forest [41] and reported the average score. We also used automated machine learning (Auto-sklearn [42]) as it employs an ensemble of top performing models discovered during the optimisation process and reports the best result.

Since we are interested in finding a classifier that is close to 100% recall on the real log sessions (i.e., successful in detecting all real log sessions) and a high recall on the simulated sessions (i.e., good at detecting most of simulated sessions), we incorporate a bias in the classifier by weighting the class of real data ($w_{real} = 10^4$, $w_{simulated} = 1$) as we need to penalise bad real log sessions predictions.

In order to calculate improvements of our models in comparison to the baseline, we utilise metrics such as *Precision*, *Recall*, *F-measure* and *Accuracy* which are common for objectively measuring the classifier's performance. In our case, we consider *True Positive* (TP) to be the scenario where the model classifies simulated sessions as simulated. A score of 0.5 (chance level of balanced binary classification) means that the classifier cannot distinguish between simulated and real log sessions and therefore, the simulated sessions are similar to real log data sessions. With a score below 0.5, the classifier performs even worse than chance-level, whereas with a score of 1, simulated sessions and log data are completely different.

Since we can distinguish between real log and simulated sessions, reporting the accuracy alone can obfuscate some of the performance that F-measure would highlight. F-measure tells how precise the classifier is (i.e., how many instances it classifies correctly), as well as how robust it is (i.e., does not miss a significant number of instances). In fact, if F-measure showed low precision/recall along with a low accuracy, we can have better confidence in the results. Therefore, we utilise all four metrics to demonstrate relative performance and consistency of the results.

5. Experimental Results

5.1. Kolmogorov-Smirnov-based Evaluation

For this evaluation test, we used the baseline Markov model approach described in section 3.1 to simulate global user search sessions and a separate model for each approach to simulate type-specific user search sessions. For each model, we utilise the transition probabilities between states which are drawn from the log sessions and the simulated sessions separately to generate two independent samples. By feeding these data points to the given formula (Eq. 6) , we got the results reported in Table 1. The p-value provides a measure of how much we can trust the statistical test. In general, the closest the p-value to zero is, the more reliable the test is. Likewise, the critical value for the two samples can be approximated using the following formula: $D_{n1,n2} = c(\alpha)\sqrt{\frac{n1+n2}{n1 \cdot n2}}$ where α refers to the level of significance and $n1, n2$ are observations from the first and second sample respectively. In our experiment, we set $n1 = n2 = 300,000$ and $\alpha = 0.01$.

Table 1
Two Sample Kolmogorov-Smirnov Test of log sessions and simulated sessions distributions using different Markov model approaches. The p-value provides a measure of how much we can trust the statistical test (the closest the p-value to zero is, the more reliable the test is).

Approach		**Kolmogorov-Smirnov Test**	
		D-statistic (p-value)	***D-critical***
Basic (Baseline)		0.00417 $(7.44e-11)$	0.00421
Conditional		0.00367 $(1.36e-12)$	0.00398
Contextual	Exploratory	0.00381 $(6.54e-12)$	0.00389
	known-item	0.00302 $(4.11e-12)$	0.00356
Time-aware		0.00406 $(6.57e-12)$	0.00415
Query-change		0.00387 $(3.88e-12)$	0.00391

Table 1 shows that the statistical value is smaller than the critical value across all models, hence we retain the null hypothesis. Therefore, we conclude that the simulated and the real log sessions belong to the same distribution.

5.2. Classification-based Evaluation

Since the KS-2 critical values are all significant, it means that our different approaches do not improve the simulation or at least it is hard to quantify the improvement using a KS-2 test. Therefore, we need to adopt a second evaluation method: we investigate whether we can train a classifier and try to distinguish between real log and simulated sessions through controlled scenarios. For each scenario, we simulate an equal amount of sessions as present in the log data to balance class distribution for each approach, namely, baseline, conditional, contextual, time-aware and query-change.

We investigate whether we can distinguish between log sessions and simulated sessions that were generated using our predefined approaches in section 3. Table 2 shows that conditional

Table 2
Classification of real log sessions vs simulated sessions using baseline and different Markov model approaches. We report the accuracy, recall, precision and F-measure across 10-CV folds (while (a) averaging over five classifiers defined in subsection 4.2.2 and (b) using Auto-sklearn. W.Sum is the weighted average score (with 0.39 and 0.61 being the size of exploratory and known-item models respectively) and RI is the relative improvement compared to the baseline. Bold indicates the best result in terms of the corresponding metric. Lowest results are the best as we aim to reduce the classifier's capability to distinguish between real log and simulated sessions.

Approach		**Accuracy (RI)**	**Recall (RI)**	**Precision (RI)**	**F-measure (RI)**
Basic (Baseline)		0.661	0.773	0.756	0.750
Conditional		**0.545** *(17.54%)*	0.578 *(25.22%)*	0.589 *(22.08%)*	0.583 *(22.27%)*
Contextual	Exploratory	0.608	0.527	0.636	0.568
	known-item	0.564	**0.487**	**0.573**	**0.509**
W.Sum		0.581 *(12.08%)*	0.502 *(34.98%)*	0.597 *(20.96%)*	0.532 *(29.07%)*
Time-aware		0.653 *(1.21%)*	0.764 *(1.16%)*	0.732 *(3.17%)*	0.748 *(2.67%)*
Query-change		0.551 *(16.64%)*	0.582 *(24.71%)*	0.601 *(20.50%)*	0.562 *(25.06%)*

(a) Averaging over five classifiers defined in subsection 4.2.2

Approach		**Accuracy (RI)**	**Recall (RI)**	**Precision (RI)**	**F-measure (RI)**
Basic (Baseline)		0.660	0.764	0.748	0.746
Conditional		**0.512** *(22.42%)*	0.577 *(24.47%)*	0.582 *(22.19%)*	0.579 *(22.38%)*
Contextual	Exploratory	0.612	0.531	0.631	0.577
	known-item	0.568	**0.486**	**0.577**	**0.528**
W.Sum		0.585 *(11.34%)*	0.503 *(34.09%)*	0.598 *(20.05%)*	0.546 *(26.71%)*
Time-aware		0.655 *(7.57%)*	0.742 *(2.87%)*	0.733 *(2.01%)*	0.737 *(1.21%)*
Query-change		0.536 *(18.78%)*	0.589 *(22.91%)*	0.591 *(20.98%)*	0.590 *(20.91%)*

(b) Using Auto-sklearn

Markov model, which simulate user actions based on the query-action approach, achieves higher performance in comparison to the baseline with a relative improvement of 22% for the F-measure score. This demonstrates that separating user search actions depending on the query's ordinal position help improving the simulation quality.

Table 2 also shows that when using the contextual Markov approach, the model did better while simulating sessions for *known-item* search types with an F-measure score of 0.509 in comparison to *exploratory search* search types with a score of 0.568. One possible explanation for this is that known-item sessions are probably easier to simulate since there is less variation. The exploratory group of users generate longer sessions, thus higher total of state transitions which results in a diverse number of simulated sessions. Using search types to discover context helped creating simulated sessions more similar to original ones (i.e., reducing the accuracy of the classifier). In addition, table 3a also reports low precision (i.e., 0.573 for *known-item* and 0.636 for *exploratory*)/recall (i.e., 0.487 for *known-item* and 0.527 for *exploratory*) values along with a low accuracy score (i.e., 0.564 for *known-item* and 0.608 for *exploratory*) when using the exploratory-known-item approach in comparison to the baseline, which indicates that we can have better confidence in the results.

Our initial hypothesis was that the baseline model should outperform the time-aware Markov model since the baseline is less complex to learn (e.g., excluding dwell time features). The less information needed to encode in the model the easier it is to get "similar" simulated data. However, adding time improves our model's simulation quality as we gained around 2.67%/1.21% in term of the relative improvement over the baseline. Another hypothesis that we have tested is to replace the estimated time distribution between actions using gamma distribution with a calculated average of dwell time of the same state transition and feed it as a feature to the model [43]. We concluded that the gamma estimation scored much better than averaging the dwell time value.

Although these results do not undoubtfully prove that we succeeded in simulating look-alike user search sessions, they show that users in our simulated sessions behave in a way that comes closer to the original behaviour from the log data.

6. Conclusion

In this paper we have presented various variants of Markov model-based simulation techniques that simulate global and user-type specific behaviour models. In our approaches, we first introduced a formal baseline that consists of exploring a first-order Markov model which we argue to be the best fit for our dataset. Then we extended the work of Baeza-Yates et al. [20] and presented a conditional Markov approach that is adapted for simulation modeling. Next, we initiated the problem of learning contextual Markov models where we partitioned users according to their navigation behaviour. In particular, we grouped users by learning a mixture of search characteristics that categorise them into *exploratory search* and *known-item* search types. Then we explored the work of Huang et al. [29] and adopted user's session-level query change to develop a syntactic Markov model that models the syntactic change which occurs between two consecutive queries. Finally, in our time-aware approach, we modeled the dwell time that users spend in between actions using the gamma distribution and then fed it as a feature to our evaluation model. There are several straightforward extensions to our work. We can use better clustering techniques for context discovery in contextual Markov models. Another extension is to model the duration of state transition differently.

We performed experiments on a real-wold dataset with conditional, contextual, time-aware and query-change Markov models and provided more empirical results showing that our Markov approaches succeeded in simulating sessions that are deemed to be good approximations of actual user sessions, making our Markov models approaches a suitable choice for user session simulation.

Acknowledgments

This work has been partially carried out within the project "SINIR: Simulating INteractive Information Retrieval" funded by the DFG (grant HA 5851/3-1).

References

[1] H. Xie, D. Wolfram, State digital library usability: Contributing organizational factors, Journal of the American society for information science and technology 53 (2002) 1085–1097.

[2] Y. Zhang, X. Liu, C. Zhai, Information retrieval evaluation as search simulation: A general formal framework for ir evaluation, in: Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, 2017, pp. 193–200.

[3] I. Cadez, D. Heckerman, C. Meek, P. Smyth, S. White, Visualization of navigation patterns on a web site using model-based clustering, in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000, pp. 280–284.

[4] P. L. Pirolli, J. E. Pitkow, Distributions of surfers' paths through the world wide web: Empirical characterizations, World Wide Web 2 (1999) 29–45.

[5] E. Manavoglu, D. Pavlov, C. L. Giles, Probabilistic user behavior models, in: Third IEEE International Conference on Data Mining, IEEE, 2003, pp. 203–210.

[6] P. Haider, L. Chiarandini, U. Brefeld, Discriminative clustering for market segmentation, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012, pp. 417–425.

[7] Y. Lucas, P.-E. Portier, L. Laporte, L. He-Guelton, O. Caelen, M. Granitzer, S. Calabretto, Towards automated feature engineering for credit card fraud detection using multi-perspective hmms, Future Generation Computer Systems 102 (2020) 393–402.

[8] A. Krogh, B. Larsson, G. Von Heijne, E. L. Sonnhammer, Predicting transmembrane protein topology with a hidden markov model: application to complete genomes, Journal of molecular biology 305 (2001) 567–580.

[9] Y. Xie, S.-Z. Yu, A large-scale hidden semi-markov model for anomaly detection on user browsing behaviors, IEEE/ACM transactions on networking 17 (2008) 54–65.

[10] I. Cadez, D. Heckerman, C. Meek, P. Smyth, S. White, Visualization of navigation patterns on a web site using model-based clustering, in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000, pp. 280–284.

[11] Y. Liu, B. Gao, T.-Y. Liu, Y. Zhang, Z. Ma, S. He, H. Li, Browserank: letting web users vote for page importance, in: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, 2008, pp. 451–458.

[12] L. Azzopardi, The economics in interactive information retrieval, in: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, 2011, pp. 15–24.

[13] X. Dongshan, S. Junyi, A new markov model for web access prediction, Computing in Science & Engineering 4 (2002) 34–39.

[14] M. Deshpande, G. Karypis, Selective markov models for predicting web page accesses, ACM transactions on internet technology (TOIT) 4 (2004) 163–184.

[15] J. Borges, M. Levene, Evaluating variable-length markov chain models for analysis of user web navigation sessions, IEEE Transactions on Knowledge and Data Engineering 19 (2007) 441–452.

[16] Y. Lucas, P.-E. Portier, L. Laporte, L. He-Guelton, O. Caelen, M. Granitzer, S. Calabretto, Towards automated feature engineering for credit card fraud detection using multi-

perspective hmms, Future Generation Computer Systems 102 (2020) 393–402.

[17] J. Kiseleva, H. Thanh Lam, M. Pechenizkiy, T. Calders, Discovering temporal hidden contexts in web sessions for user trail prediction, in: Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 1067–1074.

[18] D. Budd, L. C. Burns, Z. Guo, G. L'Italien, P. Lapuerta, Impact of early intervention and disease modification in patients with predementia alzheimer's disease: a markov model simulation, ClinicoEconomics and outcomes research: CEOR 3 (2011) 189.

[19] M. H. Davis, Markov models and optimization, Routledge, 2018.

[20] R. Baeza-Yates, C. Hurtado, M. Mendoza, G. Dupret, Modeling user search behavior, in: Third Latin American Web Congress (LA-WEB'2005), IEEE, 2005, pp. 10–pp.

[21] G. Marchionini, Exploratory search: from finding to understanding, Communications of the ACM 49 (2006) 41–46.

[22] K. Athukorala, D. Głowacka, G. Jacucci, A. Oulasvirta, J. Vreeken, Is exploratory search different? a comparison of information search behavior for exploratory and lookup tasks, J. Assoc. Inf. Sci. Technol. 67 (2016) 2635–2651. URL: https://doi.org/10.1002/asi.23617. doi:10.1002/asi.23617.

[23] J. Mun, Understanding and choosing the right probability distributions, Adv. Anal. Model (2015) 899–917.

[24] A. Hassan, R. Jones, K. L. Klinkner, Beyond dcg: user behavior as a predictor of a successful search, in: Proceedings of the third ACM international conference on Web search and data mining, 2010, pp. 221–230.

[25] R. V. Hogg, J. McKean, A. T. Craig, Introduction to mathematical statistics, Pearson Education, 2005.

[26] J. Jiang, C. Ni, What affects word changes in query reformulation during a task-based search session?, 2016, pp. 111–120. doi:10.1145/2854946.2854978.

[27] H. Yang, D. Guan, S. Zhang, The query change model: Modeling session search as a markov decision process, ACM Trans. Inf. Syst. 33 (2015). URL: https://doi.org/10.1145/2747874. doi:10.1145/2747874.

[28] M. Sloan, H. Yang, J. Wang, A term-based methodology for query reformulation understanding, Information Retrieval Journal 18 (2015) 145–165. URL: http://dx.doi.org/10.1007/s10791-015-9251-5. doi:10.1007/s10791-015-9251-5.

[29] J. Huang, E. N. Efthimiadis, Analyzing and evaluating query reformulation strategies in web search logs, in: Proceedings of the 18th ACM conference on Information and knowledge management, 2009, pp. 77–86.

[30] D. S. Hirschberg, Algorithms for the longest common subsequence problem, J. ACM 24 (1977) 664–675. URL: https://doi.org/10.1145/322033.322044. doi:10.1145/322033.322044.

[31] P. Mayr, Sowiport user search sessions data set (suss), GESIS Datorium (2016).

[32] P. Billingsley, Statistical methods in markov chains, The annals of mathematical statistics (1961) 12–40.

[33] R. N. Hiscott, Chi-square tests for markov chain analysis, Journal of the International Association for Mathematical Geology 13 (1981) 69–80.

[34] T. W. Anderson, L. A. Goodman, Statistical inference about markov chains, The Annals of Mathematical Statistics (1957) 89–110.

[35] F. J. Massey Jr, The kolmogorov-smirnov test for goodness of fit, Journal of the American statistical Association 46 (1951) 68–78.

[36] J. Heaton, An empirical analysis of feature engineering for predictive modeling, SouthEastCon 2016 (2016). URL: http://dx.doi.org/10.1109/SECON.2016.7506650. doi:10.1109/secon.2016.7506650.

[37] D. G. Kleinbaum, K. Dietz, M. Gail, M. Klein, M. Klein, Logistic regression, Springer, 2002.

[38] W. S. Noble, What is a support vector machine?, Nature biotechnology 24 (2006) 1565–1567.

[39] S. A. Dudani, The distance-weighted k-nearest-neighbor rule, IEEE Transactions on Systems, Man, and Cybernetics (1976) 325–327.

[40] J. R. Quinlan, Simplifying decision trees, International journal of man-machine studies 27 (1987) 221–234.

[41] M. Belgiu, L. Drăguţ, Random forest in remote sensing: A review of applications and future directions, ISPRS journal of photogrammetry and remote sensing 114 (2016) 24–31.

[42] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, F. Hutter, Auto-sklearn: efficient and robust automated machine learning, in: Automated Machine Learning, Springer, Cham, 2019, pp. 113–134.

[43] V. Tran, D. Maxwell, N. Fuhr, L. Azzopardi, Personalised search time prediction using markov chains, in: Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, 2017, pp. 237–240.