Search Engine Switching Detection Based on User Personal Preferences and Behavior Patterns

Denis Savenkov, Dmitry Lagun, Qiaoling Liu
Mathematics & Computer Science
Emory University
Atlanta, US
{denis.savenkov, dlagun, qiaoling.liu}@emory.edu

ABSTRACT

Sometimes, during a search task users may switch from one search engine to another for several reasons, e.g., dissatisfaction with the current search results or desire for broader topic coverage. Detecting the fact of switching is difficult but important for understanding users' satisfaction with the search engine and the complexity of their search tasks, leading to economic significance for search providers. Previous research on switching detection mainly focused on studying different signals useful for the task and particular reasons for switching. Although it is known that switching is a personal choice of a user and different users have different search behavior, little has been done to understand how these differences could be used for switching detection. In this paper we study the effectiveness of learning personal behavior patterns for switching detection and present a personalized approach which uses user's session history containing sessions with and without switches. Experiments show that users' personal habits and behavior patterns are indeed among the most informative signals. Our findings can be used by a search log analyzer for engine switching detection and potentially other log mining problems, thus providing valuable signals for search providers to improve user experience.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

search engine switching

1. INTRODUCTION

Search engines such as Google, Bing, Yandex, etc. facilitate access to enormous amount of information available on the Word Wide Web. Among many available search engines, users typically prefer to use one or two, performing majority of their searches on a primary search engine. Decision on which search engine to use as a primary search provider could be based on many factors including search quality, locale, satisfaction, usability of search interface and popularity of a search engine [9]. Sometimes users switch from one search engine to another during a search task. As the majority of switching cases (57%) arise from users' dissatisfaction with the search engine [9], information on when users switch provides a valuable signal for search providers to improve user search experience in such cases. For some search sessions, the fact of switching can be easily detected, for instance via a web browser (maintained by a search engine), a browser toolbar or search logs (e.g. some users ask navigational query in the current search engine to open a new one). However, in reality only a fraction of switches can be monitored. The same users may switch in a way that cannot be detected by a search engine, e.g. by opening a new tab. Detecting the fact of switching, when such events cannot be monitored directly is difficult, but it is important for understanding users' satisfaction and the complexity of search. It also has the economic significance to search providers as switching is a low-level signal related to the market share.

Previous work [20, 21, 9] identified the most popular reasons for switching as dissatisfaction with search results, desire to get more relevant results for coverage and verification or users' personal preferences to use a particular search engine for some types of searches. This suggests that decision to switch search engines depends not only on a task, but also on user's personal behavior patterns and habits. For example, some users are more persistent than others and can make more effort exploring the search results. Previous research made progress in detecting search engine switching, but information on user's typical behavior didn't get enough attention.

In this paper, we focus on learning users' search behavior patterns and habits and using it for search engine switching detection. Our experiments show, that a state-of-the-art existing model (semi-supervised generative model proposed in [10]) benefits from using this information. In addition, we built a model based on multiple gradient boosting regression trees, which utilizes more than 400 features based on user preferences, search tasks, user behavior patterns, etc.

In our work we used publicly available dataset, which contains search logs for 1 month period, collected by Yandex for the Switching detection challenge[1] (Oct 23, 2012 - Dec 22, 2012) . The logs have search engine switching actions detected either by a browser toolbar or by clicks on URL of

SIGIR'13, July 28–August 1, 2013, Dublin, Ireland.

[1] http://switchdetect.yandex.ru/en

a different search engine from the result page of the source search engine.

The main contributions of this paper include:

- a personalized version of the generative model for web search success prediction [10]. We show that learning user's personal action transition patterns significantly improves the performance of the model.
- machine learning based switching prediction model, based on an ensemble of decision trees, which utilizes over 400 features, including users' personal search behavior information. We show that the personalized model outperforms algorithms that do not use this information. This model outperformed other competitors on the Yandex switching detection challenge.
- in-depth feature importance analysis, which shows that users' statistics are actually among the best predictors in the feature set.

The rest of the paper is organized as follows: Section 2 briefly discusses the related work, Section 3 describes the dataset we used and explains why personalization is important. Section 4 presents our personalized versions of the recently proposed Markov model based algorithm [10] and our machine learning based approach that harnesses rich personalized behavior features for switching detection. In Section 5 the results of the prediction experiments are presented and Section 6 provides some discussions and implications of the results. Finally, Section 7 concludes the paper.

2. RELATED WORK

Understanding and detecting search engine switching has recently attracted much research attention [13][14][20][9]. Heath and White [13] proposed the task of predicting search engine switching in a real-time setting, i.e. whether a user's next action during a search will be an engine switch. The algorithm used in the paper is based on a string matching idea, i.e., user's most recent actions performed and pages visited in the session so far are encoded as a character sequence and prediction is based on the number of times this sequence was previously followed by a switch or a non-switch action. Laxmant et al. [14] later improved the idea by identifying frequent predictive subsequences and estimating their associated Hidden Markov Models. Since a sequence could not catch all session information, White and Dumais [20] used a richer set of features to predict switching, which were derived from the user's most recent query, session interaction observed so far, and user search history. Guo et al. [9] investigated the reasons causing search engine switching based on in-situ feedback from users and developed models to predict the rationale for a given switching event. The above work (especially [13][14][20]) are the closest to ours, which also inspired us on the features to start with; yet several differences exist in between. First, we aim at offline detection of user switching using complete session information instead of online scenario where only partial session information is available. Second, we focus on using personalization for learning to detect search engine switching and we compare different personalization approaches, which have been little studied in previous works.

As the most popular switching rationale is dissatisfaction with search results, another relevant research area is predicting search success [10][1][8][11][12]. Field et al. [8] worked on predicting user-reported frustration, and found a few simple query log features performed better than physical sensor features. Ageev et al. [1] worked on modeling search success via a game-like infrastructure and used Conditional Random Field (CRF) models for prediction. Hassan et al. [11][12] proposed to use two Markov models for predicting search success, and found that user behavior based models are more predictive than document relevance based models. Hassan [10] recently proposed a semi-supervised approach based on a generative model and EM estimation to model search satisfaction and found that adding unlabeled data improves the effectiveness. Other relevant work include using query logs and user behavior to predict relevance of search results [7][2] as well as studying the change in user behavior when search becomes more difficult [3]. User preference is another reason causing search engine switching, e.g. sometimes a user prefers one search engine to another for some types of search tasks. Numerous researches show that user differences influence their search strategies and behavior [5][22][17] and personalization is effective to improve user experience in web search [4][18] [6].

In our work, we take a close look at the main reasons for engine switching, derive features shown useful for identifying such scenarios and focus on learning users' personal behavior models. The results of this work show that such models significantly improve the performance of switching detection and the approach can be applied to other similar tasks.

3. DATASET ANALYSIS

3.1 Dataset

For our work we used a publicly available dataset, provided by Yandex for its recent switching detection challenge. The dataset contains a subset of search logs of 30 days, which are about 1.5 years old and do not contain sessions with queries that have commercial intent detected with Yandex proprietary query classifier. Search sessions contain unique user identifier and a sequence of records for search actions, such as queries, result clicks and search engine switching actions, which were detected by a browser toolbar or by clicks on a link to open another search engine from the search engine results page. The information provided along with search actions includes for each record the time elapsed from the beginning of the session, for a click action the clicked url, or for a query action the query and the urls shown to the user. To allay privacy concerns the logs were fully anonymized, so only meaningless query, user and url ids are available. Also the dataset doesn't contain any information on the search engine the user switched to. The elapsed time for all actions are specified in time units with unknown relation to seconds. In total, the dataset includes 8,595,731 sessions, 1,457,533 sessions with switch action, 10,139,547 unique queries, 49,029,185 unique URLs, and 956,536 unique users. According to the challenge setting, the data in the last 3 days are used as the test set, and all users who don't have switches in the first 27 days were removed from the dataset.

3.2 Switching Behavior Analysis

Previous research [20] [9] showed that the main reasons for search engine switching are: dissatisfaction in 57% of the cases, verifying or finding additional information in 26% of the cases and user preferences (e.g. a user prefers one engine to another for some particular search tasks) in 12% of

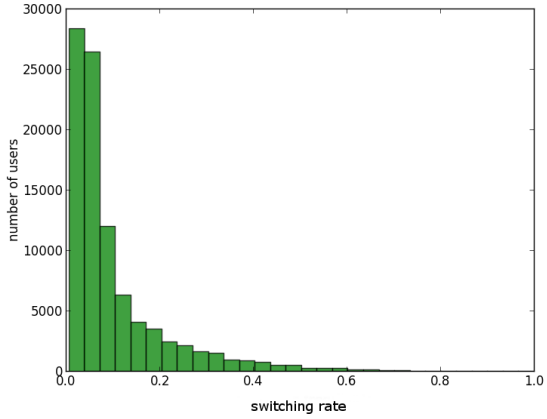

Figure 1: Histogram of switching rates for users with $\geq$ 20 sessions in the first 27 days

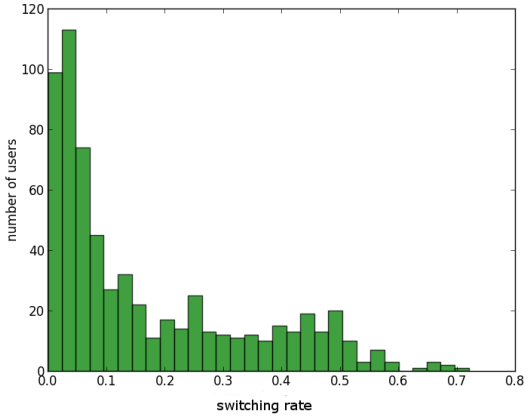

Figure 2: Histogram of switching rates for users with $\geq$ 20 sessions in the first 27 days, considering only sessions with 7 to 30 queries

the cases. This suggests that switching behavior highly depends on a search task, e.g. for an easy task the user is less likely to switch to another search engine. In [20] a histogram of probability of switching given session length is provided, suggesting that search sessions with more queries are more likely to contain a switch. But on the other hand switching behavior also depends on users' personal habits. Some users might not switch search engines even when they are not satisfied with the search results while some other users might have a habit of verifying results by opening another search engine even when the current results are relevant. Figure 1 shows the histogram of switching rates for users with at least 20 sessions in the training period (first 27 days). As you can see, some users switch more often than others. To reduce the influence of task difficulty on switching probability, the histogram on Figure 2 considers only sessions with 7 to 30 queries, as for these sessions the probability of switching is relatively more stable. Nevertheless, the histogram shows that even in this case different users still have varying switching rates. This supports the hypothesis that switching behavior have a strong personal component. Some users are more experienced with using multiple search engines, and others do this only occasionally. Another aspect relates to the search task itself. For example, a user may switch more frequently because he/she performs difficult searches more often, for which the results are more likely to be non-relevant, or because he/she asks more queries of research type and needs more search results. This means that personalization can help learn more about both users' behavior patterns and their search tasks.

As shown by White and Dumais [20], the likelihood that a switch happens in a session increases as the session length increases. We also found this result with our data by comparing the average number of queries in switching and non-switching sessions across all users (the blue line in Figure 3 shows that on average more queries are issued in sessions with a switch). However, another interesting observation is that this effect is actually different for different users. As a session gets longer, the likelihood of switching increases much more for some users than for others. Moreover, for some users the switching likelihood even decreases when more queries are issued (the left gray part in Figure 3). Therefore, to better detect switching for these users it is necessary to understand and model their personal behavior patterns.

3.3 Task and Metric Description

In this paper we are focusing on detecting search sessions that contain switch/switches to another search engine. Following the setting of Yandex switching detection challenge, we use the AUC (Area Under Curve) measure [15] to evaluate the performance of our switching prediction models. AUC is a popular measure used for evaluating classifiers, which represents the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. To allow easier interpretation of the performance of our models, we also present the corresponding precision-recall curves in the evaluation.

4. PERSONALIZED SWITCH MODELING

4.1 Evaluation Set

The dataset contains sessions collected for 30 day period from users who switched at least once in the first 27 days. Since sessions in days 28-30 were used as the test set in Yandex switching detection challenge and the labels were not available at the time of our experiments, we held out the sessions in days 25-27 as our validation set. The rest of the data (sessions in days 1-24) were used for training. To make a validation set similar to the challenge test set, we filtered out sessions from users who didn't switch in the first 24 days.

4.2 Personalized Generative Models to Switch Detection

Search trails, i.e. sequences of actions a user performs with the search engine (e.g. queries, clicks on a search result) were shown to be very useful signals for predicting user satisfaction and search engine switching. For example, in [20] pre- and post-switch motifs were studied and shown to be useful for switching prediction. In [10] a semi-supervised generative model for search success prediction was proposed, which learns probabilities of transitions from one action to

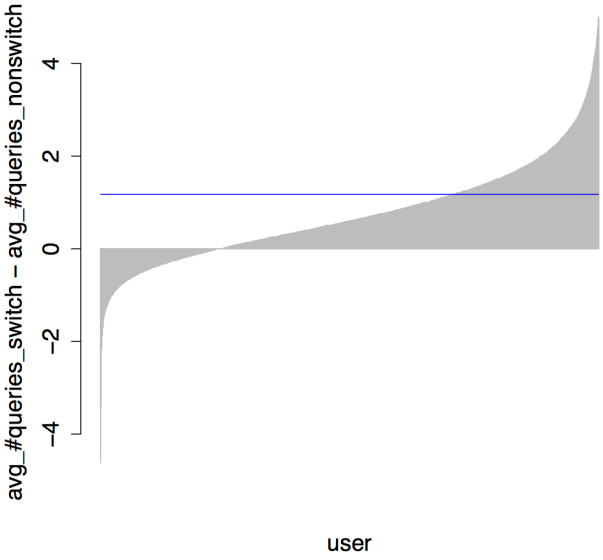

Figure 3: Difference between average number of queries in switch and non-switch sessions for each user

Table 1: Alphabets for search trails (type-I, type-II)

Action	Description
Q	query submission
C	click on a search result
E	end of the session
q	query with a short pause before next action
Q	query with a long pause before next action
K	other query
D	short dwell-time click on a search result
S	long dwell-time click on a search result
P	other clicks on a search result
E	end of the session

another and uses this data for prediction. Unfortunately, in our dataset only query and search result click actions are available, thus we could not exploit other actions such as clicks on ads or on spelling suggestions as used in previous work. As a result, the basic (type-I) alphabet for search trail encoding consists of {Q, C, E} as shown in Table 1. Inspired by previous work, we also introduce an advanced search trail encoding by distinguishing between short- and long-dwell time clicks. In addition, we felt that queries with different pause until the next action may be also helpful, so we encoded them as well. Finally, the advanced (type-II) alphabet is {q, Q, K, D, S, P, E}, as shown in Table 1. Note that, "short" is defined by less than 200 time units while "long" is defined by more than 500 time units.

A key part of the model defined in [10] for user satisfaction prediction is the action transition parameters for the switch and non-switch class respectively. Therefore, we presents the estimated transition probabilities for switch and non-switch sessions in Table 2. As you can see sessions with switches are more likely to end after a query action than sessions without switches (Note that switch actions were only used to split

Table 2: Markov model transition probabilities

	Non-switch sessions				Switch sessions		
	Q	C	E		Q	C	E
Q	0.217	0.750	0.033	Q	0.316	0.621	0.063
C	0.207	0.470	0.323	C	0.288	0.521	0.191

Table 3: Markov model transition probabilities for a particular user

	Non-switch sessions				Switch sessions		
	Q	C	E		Q	C	E
Q	0.417	0.527	0.056	Q	0.506	0.430	0.065
C	0.219	0.352	0.428	C	0.397	0.354	0.249

sessions into the two classes, not considered in transitions). Moreover, in switch sessions users tend to ask more queries.

However, we were also aware that some users have very different behavior from the average. For example, Table 3 shows the transition probabilities estimated on sessions from a particular user (who has 50 switching and 77 non-switching sessions). The differences in the estimated probabilities for this user and the general model (e.g., this user in general asks more queries) suggests that taking personal transition habits into account may improve the prediction performance.

Unfortunately, most users have just a few sessions and hence the estimated transition models can be unreliable. In this consideration, we smoothed each personalized model with the global model for final prediction. To get a good smoothing, we used a machine learning approach. Three features were generated for each session: prediction of the global model, prediction of the user's personal model, and number of sessions in the history of the user. Then a logistic regression was applied to get a combined prediction[2].

The results of our experiments with generative models are presented in Table 4. The personalized models significantly outperform the global models and other baselines. In our experiment, the semi-supervised approach proposed in [10] incorporating the EM algorithm did not improve the performance of the supervised models, perhaps due to the large amount of labeled data available. Unfortunately, encoding time information into action sequences did not help the personalized models, probably because the larger alphabet makes the action transitions more sparse and therefore the derived personalized model more unreliable.

Another interesting observation is the performance of the baseline using user switching rate compared to other baselines. Despite the fact that this predictor did not use any information about the session, it outperforms other baselines, supporting our hypothesis that switching highly depends on users' habits and behavior patterns.

4.3 Personalized Machine Learning Approach to Switch Detection

Machine learning approaches have been shown to be successful for predicting search engine switching [13][14][20] as it allows to combine various signals into a single model. Inspired by these efforts, we developed a broad set of features that cover different aspects of search behavior, including the ones described in previous work and some new ones.

[2]We also tried other models, including decision trees, but in this setup the results of different models did not differ much.

Table 4: Generative model search prediction results

Model	AUC
Baseline using query length	0.6710
Baseline using session duration	0.7257
Baseline using user switching rate (smoothed)	0.7306
Generative model (type-I sequences)	0.7058
Generative model (type-II sequences)	0.7081
Semi-supervised model (type-I sequences)	0.7064
Prediction of per-user model only (type-I)	0.6707
Personalized generative model (type-I)	**0.7725**
Personalized generative model (type-II)	0.7624

To build a personalized model we tried several approaches:

- training a separate model for each user and using its prediction as a new feature for a global training;
- encoding information about the user directly into the feature set as binary "user id" features (the total number of features becomes very large);
- calculating different statistics based on each user's sessions and using these statistics in the feature set separately as well as for normalizing other features (e.g. query clickthrough rate normalized by the average query clickthrough rate for this user). Statistics can be further calculated for users' switch and non-switch sessions separately to learn more about user behavior in both cases. Yet we need to use a separate set from the training set for computing such statistics to avoid overfitting. Therefore, we splitted the sessions from days 1-24 into two sets: a statistics set (sessions from days 1-21) and a training set (sessions from days 22-24). Then, from the training set we removed sessions by users who never switched during the statistics period, similarly as how the test set was created in the challenge setting.

4.3.1 Feature Description

The feature set we developed includes 414 features. Some of them are very similar to each other, only with different smoothing, normalization or other variations. Table 5 provides a summary of the features (not a complete list). The Python code generating the features is available online[3].

The features belong to one of the three groups:

1. features based on the current session only, e.g. session duration (in time units, in queries and in clicks), number of unique and abandoned queries (queries without clicks), average/min/max click dwell time and pause between actions (pause between issuing a query and the next action was also considered), average click position, number of SAT/DSAT clicks (as defined in Table 1), and some more.
2. features based on statistics computed from all switch/non-switch sessions in the statistics period.
3. features based on statistics computed from user-specific switch/non-switch sessions in the statistics period.

The later two groups of features are calculated on the statistics dataset. They can be grouped into three subsets:

- all features in group 1 normalized by the corresponding average statistics calculated across all switching/non-switching sessions (features in group 2) and across user-specific switching/non-switching sessions (features in group 3). This gives us four differently normalized versions for each feature in group 1.
- switching probability of a user/query/url. Different users have different switching probabilities. The same is true for different search tasks. To include search task related switching probability we considered queries and urls separately. For each query we calculated the number of times the query was issued before (immediately or earlier in a session) a switch in a session.
- features based on action sequences or search trails, including probabilities of this sequence appearing in switch/non-switch sessions, probabilities of this sequence under the Markov model with transition probabilities estimated on switch/non-switch sessions, and finally statistics based on all n-grams (2,3,4-grams) of this sequence, namely average ratio of their frequencies in switch and non-switch sessions.

4.3.2 Learning Algorithms

For our experiments we used the logistic regression algorithm (scikit-learn[4] implementation [16]) and the gradient boosting tree algorithm (pGBRT[5] implementation [19]).

5. EVALUATION

5.1 Personalization Performance

To test the performance we used the holdout validation set, which covers the sessions in days 25-27. Sessions in the previous 3 days (days 22-24) were used for training and the rest of the dataset (days 1-21) was used to calculate the aggregated statistics[6].

The results of the personalization approaches described in Section 4.3 are presented in Table 6. When building a model per user in the first approach we used the logistic regression (with L1-regularization and parameter C=1.0). In the second approach the number of features becomes very large and we were unable to train a gradient boosting model, so the result reported was also obtained by training a logistic regression model. In the third approach a gradient boosting tree algorithm was applied (with 400 iterations, maximum tree depth = 5 and learning rate = 0.1). For comparison purposes we also provided the performance of the logistic regression model in the same setting.

The first and second personalization methods (per-user model prediction as a feature and user ids as features) are prone to overfitting. The dataset contains a lot of users with just a few sessions (which is also true in general) and thus such models do not have good generalization properties. On the other hand, the third approach which utilizes statistics aggregated over user-specific switch and non-switch sessions significantly outperforms the other two. This model is also easier to implement in practice. It does not require building a separate model whenever a new user arrives and not

[3]http://mathcs.emory.edu/∼dsavenk/switch_detect/

[4]http://scikit-learn.org/

[5]http://machinelearning.wustl.edu/pmwiki.php/Main/Pgbrt

[6]Increasing the size of the training set by reducing the statistics period did not improve the performance of the models.

Table 5: Features used for switching detection (not a complete list)

Features based on current session only
q_count - number of queries
c_count - number of clicks
time_to_1click - time to first click in a session (or large value if no clicks)
q_abandoned - number of abandoned queries
dwell - average, max, min dwell times of clicks
duration - session duration (in time units and in actions - queries and clicks)
pause - mean, min, max pause between actions in session
unique_queries - number of unique queries issued in a session
(d)sat_clicks - number of sat/dsat clicks
ave_click_pos - average click position (11 if no clicks)
time_between_clicks - average time between clicks
last_action - is the last action a query or a click
Features based on all switch/non-switch sessions
all above features normalized by the corresponding average statistics calculated across all switching/non-switching sessions
q_switch_freq - frequency of a query occuring before switch in a session (max, mean, min over queries in a session)
q_ctr - ctr (sat, dsat, last) of a query (max, mean, min over queries)
q_ave_clickpos - average click position for a query (max, mean, min over queries)
q_freq - query frequency (max, mean, min over queries)
clicked_url_ctr - ctr (sat, dsat, last) of a clicked url (max, mean, min over all clicked urls in a session)
url_switch_freq - frequency of a clicked url apperaring before switch in a session (max, mean, min over all clicked urls)
markov_models - probability of the action sequence by Markov models estimated on switch/non-switch sessions
n-grams - average ratio of frequencies of the sequence n-grams in switch and non-switch sessions
seq_prob - probability of this exact action sequence appearing in switch/non-switch sessions
Features based on user-specific switch/non-switch sessions
all session-based features normalized by the corresponding average statistics calculated across all user's switch/non-switch sessions
all average statistics of session features computed over this user's switch/non-switch sessions
user_switch_prob - number of user sessions with a switching + 1 divided by the total number of user sessions + 10
user_switch_prob_periods - similar to *user_switch_prob* but computed for each 3-day period within the statistics period
user_last(middle,toolbar,serp)_switch_prob - number of user sessions with a switch (last/middle in session or of toolbar/serp type) divided by the total number of user sessions
user_session_count - number of sessions of this user in the statistics period
ave_time_to_switch - average time to switch for this user

Table 6: Performance of different personalization approaches

Method	AUC
Overall statistics used in feature set (boosting)	0.7782
Per-user model prediction as feature	0.7753
User ids as binary features	0.7587
User statistics used in feature set (boosting)	**0.8413**
User statistics used in feature set (logistic)	0.8348

require rebuilding a model to add a new user id. Note that the gradient boosting algorithm performed better than the logistic regression algorithm in this same setting, thanks to the power of the former algorithm to catch the non-linear interactions between the features.

Figure 4 shows how the prediction quality depends on the size of user's history (number of sessions), for personalized and non-personalized approaches. We chose the best-performing third personalization method using the gradient boosting algorithm and its non-personalized counterpart (the only difference is that all features depending on user-specific statistics are ignored) for this comparison. As we can see the personalized approach improves prediction performance for all kinds of users and the more information we have the better the result we can achieve. We do not have users without history in our dataset, but even if we have just one session we can still learn something from it and improve the results. The performance of the non-personalized approach also improves as the number of sessions in user history grows. The reasons could be that users who rarely use the search engine have different behavior from users who frequently use the search engine, and that sessions by different user groups have different level of individual differences in behavior.

5.2 Individual Feature Importance

Table 7 presents the importance scores for some of the features. We chose two scores: information gain and Gini index based score calculated on the boosting trees ensemble (denoted as boosting score). Only a couple of features from each group are presented in the table. The ranks of the features in the list are computed by the boosting score.

The analysis shows that the most important signals for switch detection are sequence n-gram statistics, user switching frequency, and average click position. Interestingly, the average click position feature is ranked 3rd by the boosting score but much worse (123th) by the information gain. This feature can be an indicator of both difficult search tasks and tasks requiring more coverage. Sequence features

Table 7: Feature importance scores (ranks are computed based on the boosting score)

Rank	Feature	Information gain	Boosting score
1	*user_typeII_3gram* - a ratio of 3-gram (type-II, see above) counts in user's switch and non-switch sessions averaged over all 2-grams in a search trail	0.0350	0.0230
2	*user_switches_count* - total number of switches in user's sessions during the statistics period	0.0194	0.0222
3	*ave_click_pos* - average position of clicks in a session	0.0149	0.0219
4	*user_switch_prob* - smoothed user's switching rate in sessions from the statistics period	0.0309	0.0184
11	*typeII_3gram* - a ratio of 3-gram counts in all switch and non-switch sessions from the statistics period averaged over all 3-grams in a search trail	0.0290	0.0134
12	*time_to_1click_user_normalized* - time to the first click in a session normalized by average time to the first click in user's non-switch sessions	0.0103	0.0129
14	*abandoned_to_ave_abandoned* - number of abandoned queries normalized by average number of abandoned queries in user's non-switch sessions	0.0229	0.0114
15	*unique queries* - number of unique queries in a session	0.0204	0.0114
21	*ave_clickedurl_switchprob* - number of times a user switched after (later in a session) clicking on the given url divided by the total number of clicks on this url averaged over all clicked urls in the session	0.0100	0.0103
22	*min_pause_user_switch* - minimum pause in a session normalized by average minimum pause in user's switch sessions	0.0016	0.0101
27	*ave_time_to_switch* - average time before switching in user's sessions	0.0026	0.0092
30	*time_1click* - time to first click in a session	0.0078	0.0082
42	*sat_clicks* - number of sat clicks in a session	0.0046	0.0069
51	*user_ave_query_switchprob* - number of times the user switched after issuing the given query divided by the total number of times the query was asked averaged over all queries in the session	0.0013	0.0059
64	*session_duration* - duration of the session in time units	0.0182	0.0050
291	*query_nonswitchprob* - number of time a query was asked and there was no switch afterwards in the session divided by the total number of times the query was asked averaged over all queries in a session	0.0001	0.0000

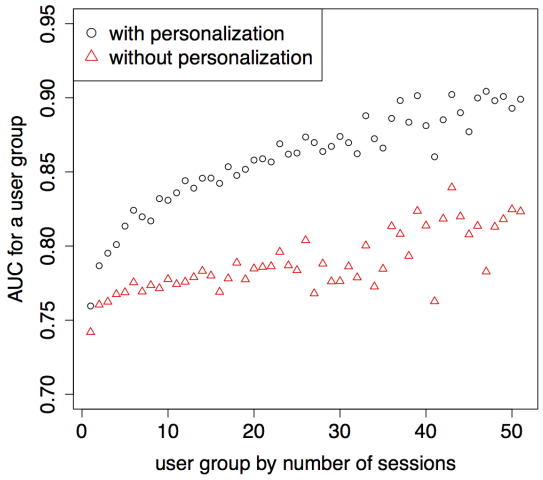

Figure 4: AUC for users with different size of search history (number of sessions)

placed at the top of the ranked list, suggesting that Markov models, whole sequence frequencies and n-gram statistics are all useful. Comparing the two types of sequences, we found that features calculated based on type-II sequences are ranked best, which means that encoding different dimensions of time information (i.e., distinguishing different lengths of pause after a query and of pause after a click separately) is useful, which supports the previous researches. It was surprising that url switching statistics (probability that a session contains a switch after a click on the given url) are more important than query switching statistics. The possible explanation is that some urls "suggest" users to try another search engine. Unfortunately we couldn't explore this deeper because the dataset is anonymized and actual urls are not available. Another reason could be sparsity of the statistics: only $\sim$ 48 % of the queries shown in sessions from the validation period were also asked at least once in the training/statistics period, and $\sim$ 67% of urls clicked during the validation period were also clicked at least once during the training period.

As the analysis showed that n-gram features are among the most useful predictors, we decided to look at some n-grams in more details. Table 8 shows some 3-grams over type-II sequences ranked by the ratio of their frequencies in switch and non-switch sessions. We can see that 3-grams with Q (queries with long pause) tend to occur more frequently in switch sessions, and on the contrary 3-grams with S (SAT clicks) are indicators of non-switch sessions.

5.3 Feature Group Importance

To investigate how features interact with each other we analyzed the prediction performance for each of the feature groups described in Chapter 4.3.1. A special interest is to see which of the following is the most important: frequency of switches for users, queries, urls or search trails. Therefore, we considered another way to group features into five dis-

Table 8: 3-grams ranked by the ratio of frequencies to appear in switching and non-switching sessions

3-gram	Ratio	3-gram	Ratio
QKQ	1.6332	qSS	0.2096
KQQ	1.5983	SSS	0.2107
QQQ	1.5895	SSP	0.2449
QQK	1.5259	SPS	0.2568
KQK	1.4143	SSD	0.2649
KKQ	1.4041	PSS	0.2697
...	...	...	...
qqq	0.4384	DDD	0.4194

joint sets: 1) features based on session statistics[7] (duration, number of queries, number of clicks, etc); 2) features based on user statistics[8] (user switch probability and some modification); 3) features based on query statistics (frequency of switches after a given query, ctr of queries in a session, etc); 4) features based on url statistics (frequency of switches after a given url is clicked in a session, ctr of clicked urls in a session, etc); and 5) features based on action sequence statistics (Markov model probabilities, n-gram statistics, etc).

Based on the above two ways of feature grouping, we conducted a series of feature ablation experiments. Table 9 shows the prediction performance when providing a single feature group or when providing all feature groups except the given one. Among the first set of feature groups, overall statistics features alone are more effective than session features alone, which verifies our intuition that aggregated statistics on a separate period helps with switching prediction. Moreover, user-specific statistics features gave even more signals for predicting switches. This single feature group run achieved 98.1% of the validation AUC obtained by training on all features. And removal of this feature group resulted in a significant drop in prediction quality (from 0.8413 to 0.7782).

Among the second set of feature groups, the most effective ones are session statistics features and sequence features. Here we also see that url switching statistics features alone obtained better AUC than user statistic features alone (e.g. user switching frequency). Query statistic features alone resulted in significantly worse performance in predicting switches. On the other hand, results of runs with a feature group removed are very close to each other, indicating that signals in each of the five feature groups could be covered to a certain extent by other four feature groups.

In Figure 5 we plotted the precision and recall curve for the positive class (switch sessions) for models with different subsets of features. It shows that for the switching detection task, high precision (e.g., 0.78) could be achieved at the cost of recall (e.g., 0.1). Also the figure suggests that adding personalized statistics improves the prediction quality significantly as you can see a gap between curves with and without personalized statistics.

5.4 Model Tuning

Considering the training set is imbalanced (only 10% of the sessions contain switches and the rest are sessions without a switch), we tried two approaches to balance the dataset, namely increasing the weight of positive examples and subsampling the negative examples. The results for both approaches are summarized in Table 10. In addition, to reduce the variance of the prediction model, an averaging approach was applied. Models for seven different time splits (e.g., days 1-3 as training period while days 4-24 as statistics period, days 4-6 as training period while days 1-3,7-24 as statistics period, etc) were built and average of the individual model predictions was used as the final score. As the results show, these strategies improve the prediction performance.

[7]Note that session statistics features here included all the normalized versions by the corresponding average statistics computed in the statistics period.

[8]Note that user statistics features here did not include those relying on queries, urls, or action sequences in the session.

Table 9: Feature ablation experiments

Feature group	Training AUC	Validation AUC
Single feature group runs		
Session features	0.7563	0.7491
Overall statistics features	0.7853	0.7777
User-specific statistics features	0.8381	0.8253
Session statistics features	0.8298	0.8183
User statistics features	0.7503	0.7273
Query statistics features	0.6480	0.6352
Url statistics features	0.7488	0.7396
Sequence statistics features	0.8062	0.7952
Without feature group runs		
Session features	0.8532	0.8326
Overall statistics features	0.8371	0.8290
User-specific statistics features	0.7882	0.7782
Session statistics features	0.8453	0.8348
User statistics features	0.8490	0.8289
Query statistics features	0.8529	0.8380
Url statistics features	0.8466	0.8273
Sequence statistics features	0.8517	0.8397
All features	0.8534	0.8413

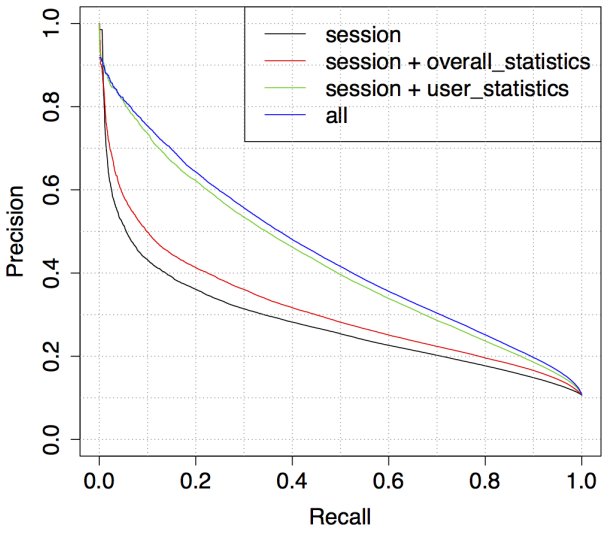

Figure 5: Precision-Recall curve for the positive class (switch sessions)

Table 10: Results for dataset balancing and model averaging

	Train AUC	Validation AUC
No balancing	0.8473	0.8403
Pos weight = 4	0.8534	0.8413
Subsampling	0.8490	0.8412
Model trained on d22-24	0.8534	0.8413
Averaging 8 models	-	**0.8450**

5.5 Comparison with Other Models

As baselines we decided to use the following algorithms: ranking by session length in queries, by session duration in time and by user switching rate on a separate period. Table 11 compares results of our model with other known algorithms. Unfortunately, some of the features described in [10] and [20] cannot be calculated base on our fully anonymized dataset, thus we included just a subset of the described features. Most of them were already a part of the feature set used for this work. We should mention that in [20] the online switch prediction problem is solved, i.e. given a part of a session predict if the next action is a switch. This problem is different from (probably more difficult than) the offline detection of switching, and thus we were unable to get high AUC score with this online model. To train it we used gradient boosting tree algorithm (200 iterations, depth = 5 and learning rate = 0.1).

Table 11: Comparison of different switching detection algorithms

Model	AUC
Baseline using query length	0.6710
Baseline using session duration	0.7257
Baseline using user switching rate (smoothed)	0.7306
Semi-supervised model from [10]	0.7081
Personalized generative model (type-I)	0.7725
Online prediction model trained on subset of features from [20]	0.7206
Our personalized model	**0.8450**

The results in Table 11 show that the developed personalized switching prediction model outperforms other known models for the switching detection task. Another interesting observation is the performance of the baselines. On our task and dataset, session duration and user switch rate predictors show better performance than the generative model. And user switching rate is the strongest baseline, which proves that switching behavior depends highly on user habits. Yandex Switching detection challenge also proved competitiveness of the presented personalized model as this method placed 1st out about $\sim$ 100 teams.

6. DISCUSSION AND IMPLICATIONS

A primary focus of this research has been a better detection of search engine switches inside a session using personalized models. We showed that users have different behavior patterns and habits when using web search and mostly ignored personal statistics information is very useful for understanding search logs. Tables 2 and 3 show an example of differences between "average" search behavior and behavior of a particular user. There are other examples, which support the idea that users do have different search behavior and we can benefit from learning behavior of each individual user. A simple baseline model, which predicts probability of a session to contain switching action based on how frequent the given user switched before performed as good as some other more complicated models. Learning more about user's behavior can also help improve performance of existing models. As an example, we trained a personalized version of one of the recently proposed models [10] - a generative model for satisfaction prediction and applied it to switching detection. Personalized model allows to achieve 9.45% improvement over non-personalized model.

Another common approach to switching detection is a machine learning based approach. To learn how personalization would affect such models we developed a set of features which include some of previously studied signals as well as some other predictors and applied a gradient boosting tree algorithm to learn a detector for session containing switching actions. In this paper we presented 3 possible personalization approaches, namely training a model for each user as well as a global model, using user id as a feature and calculating rich aggregated statistics based on user-specific sessions as both independent features and normalizers for other features. Unfortunately the first two approaches didn't show any improvement as models became overfitted, but the last approach allowed us to get a 8.1% improvement over a similar model trained on unpersonalized feature set. Our analysis showed that features based on user's statistics are among the most important features. The features ablation experiments also support the conclusion on usefulness of user's statistics. The group of features obtained by considering statistics collected on user's previous switching and non-switching sessions performed better than the others and removal of this group leads to significant decline in prediction quality.

A possible problem with this approach is the sparsity of the statistics, as just a small fraction of users have good search history. But as Figure 4 suggests even for users with small history the model still does good prediction which is even better then for non-personalized model. The same experiment shows that the more history we have the better (we only had enough data to look at users with up to 50 sessions), but the grows in the prediction performance stabilizes after as few as 10 sessions in user's history.

We believe that the features we designed and the personalization approach we used can be useful in some other tasks, like user satisfaction prediction, relevance prediction from click logs, etc. For example, some users click on more results without paying too much attention to snippets and some may examine results longer than others. This means that the clicks of these users may weight differently when estimating results relevance.

7. CONCLUSIONS

In this paper we presented a personalized approach to search engine switching prediction. The approach allows a model to learn user's personal behavior patterns in switching and non-switching sessions and use this information to improve the prediction performance. We showed that some existing models can benefit from using personalized information, e.g. the recently proposed generative markov model for user satisfaction prediction, when applied to switching detection problem, can be improved by learning per-user

models. We also designed a machine learning approach which utilizes several kinds of personalized and aggregated statistics, collected on a period separate from the training set. All the features were designed to help learning different users' switching habits and behavior patterns as well as different reasons for switching (i.e. dissatisfaction and necessity in better coverage). We demonstrated that personalized statistics and session feature normalization allow to improve switching prediction quality significantly. We believe that our approach can also be used in other log mining tasks such as search success prediction and document relevance prediction.

8. ACKNOWLEDGMENTS

This work was supported by the NSF grant IIS-1018321, the DARPA grant D11AP00269, and the Yahoo! Faculty Research Engagement Program.

9. REFERENCES

[1] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. Find It If You Can: A Game for Modeling Different Types of Web Search Success Using Interaction Data. In *Proc. of SIGIR'11*, pages 345–354, New York, New York, USA, July 2011. ACM Press.

[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 19–26, New York, NY, USA, 2006. ACM.

[3] A. Aula, R. M. Khan, and Z. Guan. How does search behavior change as search becomes more difficult? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 35–44, New York, NY, USA, 2010. ACM.

[4] P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short- and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '12, pages 185–194, New York, NY, USA, 2012. ACM.

[5] G. Buscher, R. W. White, S. Dumais, and J. Huang. Large-scale analysis of individual and task differences in search result page examination strategies. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 373–382, New York, NY, USA, 2012. ACM.

[6] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 875–883, New York, NY, USA, 2008. ACM.

[7] G. Dupret and C. Liao. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 181–190, New York, NY, USA, 2010. ACM.

[8] H. A. Feild, J. Allan, and R. Jones. Predicting searcher frustration. In *Proc. of SIGIR'10*, pages 34–41, New York, New York, USA, July 2010. ACM Press.

[9] Q. Guo, R. White, Y. Zhang, B. Anderson, and S. Dumais. Why searchers switch: understanding and predicting engine switching rationales. In *Proc. of SIGIR'11*, pages 335–344, 2011.

[10] A. Hassan. A semi-supervised approach to modeling web search satisfaction. In *Proc. of SIGIR'12*, pages 275–284, New York, NY, USA, 2012. ACM.

[11] A. Hassan, R. Jones, and K. L. Klinkner. Beyond DCG: User Behavior as a Predictor of a Successful Search. In *Proc. of WSDM'10*, pages 221–230, New York, New York, USA, Feb. 2010. ACM Press.

[12] A. Hassan, Y. Song, and L.-w. He. A task level metric for measuring web search satisfaction and its application on improving relevance estimation. In *Proc. of CIKM'11*, pages 125–134, New York, New York, USA, Oct. 2011. ACM Press.

[13] A. Heath and R. White. Defection detection: Predicting search engine switching. In *Proc. of WWW'08*, pages 1173–1174, 2008.

[14] S. Laxman, V. Tankasali, and R. W. White. Stream prediction using a generative model based on frequent episodes in event sequences. In *Proc. of SIGKDD'08*, pages 453–461, New York, New York, USA, Aug. 2008. ACM Press.

[15] C. Ling, J. Huang, and H. Zhang. Auc: a statistically consistent and more discriminating measure than accuracy. In *International Joint Conference on Artificial Intelligence*, volume 18, pages 519–526, 2003.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] S. Shen, B. Hu, W. Chen, and Q. Yang. Personalized click model through collaborative filtering. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 323–332, New York, NY, USA, 2012. ACM.

[18] D. Sontag, K. Collins-Thompson, P. N. Bennett, R. W. White, S. Dumais, and B. Billerbeck. Probabilistic models for personalizing web search. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 433–442, New York, NY, USA, 2012. ACM.

[19] S. Tyree, K. Weinberger, K. Agrawal, and J. Paykin. Parallel boosted regression trees for web search ranking. In *Proc. of WWW'11*, pages 387–396. ACM.

[20] R. White and S. Dumais. Characterizing and predicting search engine switching behavior. In *Proc. of CIKM'09*, pages 87–96. ACM, 2009.

[21] R. White, A. Kapoor, and S. Dumais. Modeling long-term search engine usage. *User Modeling, Adaptation, and Personalization*, pages 28–39, 2010.

[22] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 21–30, New York, NY, USA, 2007. ACM.