Models of Searching and Browsing: Languages, Studies, and Applications

Doug Downey
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195
ddowney@cs.washington.edu

Susan Dumais & Eric Horvitz
Microsoft Research
Redmond, WA 98052
{sdumais,horvitz}@microsoft.com

Abstract

We describe the formulation, construction, and evaluation of predictive models of human information seeking from a large dataset of Web search activities. We first introduce an expressive language for describing searching and browsing behavior, and use this language to characterize several prior studies of search behavior. Then, we focus on the construction of predictive models from the data. We review several analyses, including an exploration of the properties of users, queries, and search sessions that are most predictive of future behavior. We also investigate the influence of temporal delay on user actions, and representational tradeoffs with varying the number of steps of user activity considered. Finally, we discuss applications of the predictive models, and focus on the example of performing principled prefetching of content.

1 Introduction

User interaction with Web search engines is a commonplace, yet complex, information-seeking process. In the course of a search session, users take various actions to accomplish their goals. They formulate and issue queries, browse results, navigate to result pages or other content, reformulate their queries, request additional results, and often mix these actions in an iterative and varying manner as they pursue information. Each step in this process is informed by both the user's prior experiences and recent history with search and retrieval, the user's current goal, and the information the user gathers as the session progresses [Bates, 1989; Rose and Levinson, 2004]. Behavioral data capturing such rich interaction is becoming available via the collection of logs of Web searching and browsing activity. This newly available data provides unprecedented opportunities for research on applications of machine learning and reasoning to better understand and support human information-seeking activities.

As a concrete example, consider the log of search activity shown in Table 1. Here, a computer user starts with a general query ("drivers"), clicks on a result ("www.windrivers.com"), and then returns to the results page (presumably unsatisfied).

Time	User Action
5:04:10 PM	Queries for **drivers**
5:04:18 PM	Clicks on result **http://www.windrivers.com/**
5:04:42 PM	Returns to results for **drivers**
5:05:10 PM	Requests second result page for **drivers**
5:06:30 PM	Queries for **ati 9550 drivers**
5:06:35 PM	Clicks on result **http://support.ati.com/ics/...**
	End of Session

Table 1: **Example of Web Search Activity**

After spending some time browsing the second page of results for "drivers," the user reformulates the query to be more specific ("ati 9550 drivers"). The user then navigates to a desired result page, and the search session ends. Activity logs such as this can be used to build *predictive* models of search behavior. A search engine capable of predicting users' future actions could streamline interaction sequences such as these, potentially offering dramatic improvements in search performance.

We report on an investigation of a large dataset of Web searching and browsing activity, collected with permission from hundreds of thousands of users. We focus on the use of machine learning to study the challenges and opportunities of building predictive models from this data.

Our contributions are as follows:

1. We introduce an expressive language for representing searching and browsing behavior. This language provides a unifying framework for discussing and analyzing models of search activity, regardless of the applications for which the models may be intended. We employ the language to describe our study of search activity in the context of other studies.
2. We explore the construction of predictive models, and measure the value of different observational variables within these models. Our experiments demonstrate that several variables have significant effects on predictive accuracy. We include an analysis of the influence of different temporal delays on the likelihood of future actions, and also consider the tradeoff between complexity and predictive power when assuming different degrees of locality of evidence.
3. We formulate and experimentally evaluate a sample ap-

plication of the predictive models, aimed at the principled prefetching of content.

Section 2 describes our language for search activity models. Section 3 gives the experimental results of our predictive models, including an analysis of the influence of different observational variables. Section 4 discusses applications of the predictive models and presents experiments on applying the models to prefetch search engine results. We discuss related work in Section 5 and conclude with a summary of the work.

2 A Model of Web Search Activity

We found it useful for reflection, comparative analysis, and communication with colleagues to compose an expressive graphical language for describing search activities.

2.1 State-Machine Representation

A model of search activity is cast in the form of a state machine representation displayed in Figure 1. In this state model, nodes represent events that occur during a user's interaction with a search engine, and edges indicate the possible transitions between events. User actions, indicated by triangles in the diagram, are events that the user initiates. Boxes within the state diagram represent states that are machine generated. Lastly, the outer plates, marked by U and S, indicate that the structures they contain are repeated once for each user (U) and search session (S), respectively. Define the set of search activity elements as $\mathbf{E} = \{U, S, R, P, q, c_r, c_{\neg r}, b, v, z\}$; the individual elements of E are defined as follows:

- U – User
- S – Session
- q – Query input to search engine
- R – Result page output by search engine
- P – Non-result page output by website
- c_r – Click on search result hyperlink
- $c_{\neg r}$ – Click on non-result hyperlink
- b – Browser-assisted navigation to previously viewed pages (*e.g.,* forward/back navigation, tabbed browsing, etc.)
- v – Visit to Web page by means other than c_r, $c_{\neg r}$, b (*e.g.* by opening a new browser, typing into the address bar, clicking a hyperlink from an external application, etc.)
- z – End of session

Doubled lines are used to indicate the session start action q, and a circle indicates the end state z.

In Figure 1, each user U executes one or more sessions S. Here a *session* is defined as a sequence of search activity directed toward a single user intent; all sessions begin with a query q. Query actions are always followed by R (the presentation of a search result page), and from the R state the user can execute any action. As a concrete example, the event sequence from the activity log in Table 1 would be expressed as $qRc_rPbRqRqRc_rPz$. We will refer to the state machine in Figure 1 as the *Search Activity Model* (SAM).

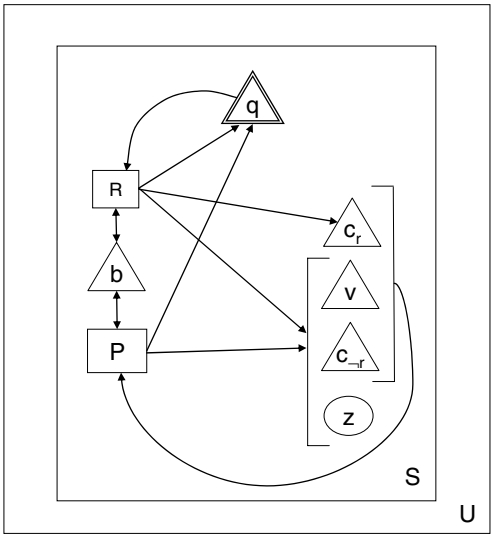

Figure 1: **Search Activity Model**

2.2 A Language for Search Activity Models

The set of event sequences generated by the full SAM is infinite; a particular model will often consider only a subset of these possible sequences. To specify the activity sequences that a particular model considers, we adopt the following definition:

Definition The *event sequence space (ESS)* of a model M is a regular expression over the vocabulary $\mathbf{E} \cup \{()_\epsilon, \tau\}$ which matches exactly those event sequences considered by M.

The added construct $()_\epsilon$ indicates events that occur, but that are not explicitly considered in the model. The added symbol τ indicates points in the sequence at which the model considers temporal duration. As an example, Lau and Horvitz [1999] investigated the influence of temporal delay on the type of query reformulation users will next perform (*e.g.,* specialization, generalization, etc.), given prior reformulations. The event sequences that the Lau & Horvitz model considers are pairs of queries separated by some number of ignored non-query events. Additionally, the model considers the duration between each pair of queries. Thus, the Lau & Horvitz model has an ESS of $q\tau(.*)_\epsilon q$. We assume that wildcards in the regular expression execute a minimal match, so a $.*$ term (which matches zero or more events of any type) is prohibited from including the event sequence following it (q in this case).

The ESS language assists with comparing and contrasting the intent and results of previous work in search activity modeling. Table 2 lists the event sequence spaces for several recent studies of search activity.[1] Note that the ESS language can be used to characterize a model regardless of the particular application for which the model may have been designed. For example, the models in Table 2 were developed for tasks

[1] An ESS always specifies a sequence of events as executed by a single user— the element U is included for cases in which users are explicitly modeled.

$q\tau(.*)_\epsilon q$	[Lau and Horvitz, 1999]
$q(R)_\epsilon c_r$	[Beeferman and Berger, 2000] [Lee *et al.*, 2005]
$qR_\epsilon(c_r P(.*)_\epsilon)+$	[Cui *et al.*, 2002]
$(qR(c_r?(.*)_\epsilon)*)+$	[Rose and Levinson, 2004]
qR	[Duame and Brill, 2004] [Davison *et al.*, 2003]
$q(.*)_\epsilon q$	[Jones and Fain, 2003] [Jones *et al.*, 2006]
$U(qR_\epsilon c_r(.*)_\epsilon)+$	[Sun *et al.*, 2005]
$S(qR(c_r?(.*)_\epsilon)*)+$	[Radlinski and Joachims, 2005]

Table 2: **Previous search models.** A summary of search activity models studied previously, and their respective event sequence spaces.

as wide ranging as query expansion, result ranking, and predicting different types of query reformulations, among others.

Importantly, all of the models in Table 2 are *server-side* models, in that the user actions they consider – q and c_r – can be detected from a search engine Web server (although c_r events must be redirected through the search engine to be logged). The full SAM is a *client-side* model, including event types (*e.g.* browser actions b) that require instrumentation installed on the user's machine. Similar client-side activity models have been investigated in recent work (*e.g.* [Agichtein *et al.*, 2006; Fox *et al.*, 2005]). With the exception of the scrolling events in [Fox *et al.*, 2005], those models can be fully expressed in our language.

A complete model of search activity is characterized by not only an ESS, but also a *parameterization*, which maps events to a vector of features (*e.g.,* a query action can be parameterized by properties like its length or average frequency). Because parameterizations can vary widely across models, we do not attempt to express them in our language. Section 3 details the particular parameterizations we employ in our experiments.

2.3 SAMlight

The full SAM considers *all* browsing activity following a search action. Because our focus is on search activity in particular, we will ignore in our experiments the browsing activity that is unlikely to be related to a search query. For example, when the user types a new URL into the browser's address bar, it is unlikely that this URL is directly relevant to a previous search action. Ignoring this browsing activity allows us to dramatically reduce the size of the data sets we analyze, while still retaining many of the events relevant to search actions. We will focus on a subset of the SAM that we call *SAMlight*. SAMlight has the following form:

$$\begin{aligned} \text{SAMlight} &= U(S(q\text{<res>}(b\text{<res>})*\,(.*)_\epsilon)+) + \\ \text{<res>} &= R_\epsilon\tau(c_r P_\epsilon \tau\text{<path>}\tau v?(.*)_\epsilon)? \\ \text{<path>} &= ((c_{\neg_r}|b)P_\epsilon)* \end{aligned}$$

SAMlight considers a set of users executing one or more sessions, where each session consists of one or more queries followed by activity surrounding a result set (<res>) generated by a search engine. Result set activity includes the presentation of a results page, followed by an optional result click. In the case of a result click, we explicitly distinguish the subsequent path (<path>) of pages traversed via hyperlink clicks or back actions,[2] which tend to be particularly related to the search query (a similar approach was taken in [Agichtein *et al.*, 2006]). We consider temporal duration at multiple points in the sequence, including the delay between all search actions, the duration of the path, and the time between the end of the path and the next action.

3 Experiments

We now turn to experiments with building predictive models of search behavior. We start by describing in detail our data collection process and the parameterization of the events we consider, as well as the machine learning methodology we employ to construct models. We then present the results of the experiments on predicting a search user's next action, and analyze which aspects of the SAM described in Section 2 are most important for the task.

3.1 Data collection and parameterization

Our data set consists of three weeks of Web activity collected via opt-in client-side instrumentation from a set of over 250,000 users. The data was collected from users who consented to share their searching activities with Microsoft via the Windows Live Toolbar download. The data consists of triples of the form (*user ID, time, URL visted*), representing user activity. We extracted each user's *search actions* (all search engine queries and result clicks) for three popular search engines as well as ten Web page requests following each search action.[3]

We partitioned the user actions into *sessions*, which, as described in Section 2, are contiguous blocks of activity over which we assume the user's search goal remains relatively constant. Detecting the boundaries of sessions is a challenging problem which could be informed by a variety of temporal and behavioral factors. For these experiments, we employed a simple heuristic that considers any search actions within 30 minutes of each other to be part of the same session, following previous work [Radlinski and Joachims, 2005]. We have explored the sensitivity of session numbers to this segmentation parameter. We found that the effect of using a shorter temporal gap of 5 minutes would have been relatively small, further dividing about 25% of the sessions.

In the models, each action is parameterized by a set of features, which is detailed in Table 3. The features express properties of different elements of the SAM, including the user, the session, query and click actions, time, and a summary of information regarding non-search actions following search

[2] Although the SAM element b refers to browser-aided navigation in general, in our experiments, we consider only the browser's "back" action.

[3] As our instrumentation returns only which URLs the user requests, we must infer when result clicks occur. We assume all non-search URL loads originating from a link on a search page are result clicks. Based on manual inspection, we estimate this heuristic to be over 95% accurate.

actions. Features describing the user (U) include statistics such as how frequently the user searches (U(qPerDay)) or the average time the user takes to click a result page (U(AvgSecToc_r)). Properties of the session (S) include features such as the number of queries issued in the current session S(Num$q$). Features of query actions include attributes of the query text (*e.g.* the number of words q(WordLen)), the query words distributions in Web text (*e.g.* the frequency of the query's least frequent word, q(MinWordFreq)), and past search behavior for the query (*e.g.* the average click position for the query, q(Avgc_rPos)). Properties of click actions include attributes like position in the result list (c_r(Position)), and whether the click was on an advertisement (c_r(IsAd)).

We used the first week of user data to estimate the "aggregate" properties of users and queries, which are averaged over multiple search sessions (*e.g.* U(Avg$S$Sec), q(Avgc_rPos)). The following week was used as a training set, and the final week as a test set. Features related to query word distributions on the Web (*e.g.* q(MinWordFreq)) were computed using a 400 million page sample of Web text.

3.2 Machine-learning methodology

The central challenge we address is that of predicting the user's next action, given a set of previous actions and information accumulated about the user and the session. Formally, we model the conditional probability distribution $P(a_n|U, S, a_{n-1}, \ldots, a_{n-k})$, where a_i represents the ith search action in the session.[4] In this distribution, actions $a_{n-1}, \ldots, a_{n-k}$ and the U and S variables are parameterized as in Table 3, and the target variable (the next action a_n) is parameterized in different ways for different experiments, as explained below.

To model the conditional probability distribution, we employed a learning method that performs Bayesian structure search to find predictive models that best explain the training data [Chickering, 2002]. The resulting models are Bayesian dependency networks in which the conditional distributions at each node take the form of probabilistic decision trees. Parameters for restricting the density of the dependency networks were estimated via cross validation on the training set.

Bayesian structure search to learn dependency networks is one of several feasible learning procedures. We initially selected the method because it scaled nicely to training sets containing half a million instances with up to hundreds of features. Also, the dependency models and trees output by the method allowed us to inspect graphical relationships among observations and predictions. A comparison of alternative machine learning algorithms (*e.g.*, logistic regression, Support Vector Machines, etc.) for this task is an item of future work.

We parameterized the target variable representing the next action at two different levels of granularity. At a *coarse* level of granularity the variable includes five output values representing the nature of the next action. We model whether the next action is a query (q), result click (c_r), or end of session

[4]Properties of the session accumulate as the session progresses; for example, S(DurationSec), when used to predict a_n, expresses the duration the session up until action a_{n-1}.

User Features (U)	
U(Avg$S$Sec)	Avg. session length (in seconds)
U(AvgSecToc_r)	Avg. search to result click interval
U(qPerSecInS)	Avg. queries per second within session
U(qRepeatRate)	Fraction of queries that are repeats
U(qPerDay)	Avg. queries per day
U(Avgc_rPos)	Avg. rank of clicked results
U(Avg$q$WordLen)	Avg. query length
U(c_rProb)	Ratio of result clicks to queries
U(PrefEngine)	Engine queried most frequently
U(PrefEngineFreq)	Fraction of queries on preferred engine
U(AvgFirstResult)	Avg. first result requested in queries
Session Features (S)	
S(Num$q$)	Number of queries issued
S(DurationSec)	Duration of the session
S(qFrac)	Ratio of queries to search actions
S(Max$q$Words)	Maximum q(WordLen) (see below)
S(Min$q$Words)	Minimum q(WordLen) (see below)
Query Features (q)	
q(WordLen)	Length in words
q(CharLen)	Length in characters
q(FirstResult)	Rank of first result requested
q(Freq)	Number of times query is issued
q(c_rProb)	Prob. some result is clicked
q(Avgc_rPos)	Avg. result position clicked
q(Avgc_rDelay)	Avg. time to click after query
q(AvgPathSec)	Avg. PathDwellSec (see below)
q(AvgPathPages)	Avg. PathPageLen (see below)
q(AvgAfterPathSec)	Avg. AfterPathSec (see below)
q(Distinct$U$)	Number of distinct users issuing query
q(HasDefinitive)	1 *iff* query has "definitive" result
q(HasSuggestion)	1 *iff* query has a spelling suggestion
q(AdImpressions)	Number of times ads shown
q(AvgNumAds)	Average number of ads shown
q(AdBid)	Average bid for ads on this query.
q(IsAdvanced)	1 *iff* query has advanced features
q(MinWordFreq)	Web frequency of least frequent word
q(MaxWordFreq)	Web frequency of most frequent word
q(GeoMeanFreq)	Geo. mean of word Web frequencies
q(AvgWordFreq)	Average of word Web frequencies
q(MaxColloqQuot)	Maximum bigram collocation quotient
q(ContainsName)	1 *iff* query contains a person name
q(ContainsLoc)	1 *iff* query contains a location name
Result Click Features (c_r)	
c_r(Position)	Result rank for c_r
c_r(DwellSec)	Time in seconds on the target page.
c_r(IsAd)	1 *iff* the click is on an advertisement
$q\|c_r$(Engine)	Search engine for query or click
Non-search Action Features ($v\|c_{\neg r}\|b$)	
PathPageLen	Length in pages of *path* after result click traversed by links or "back" actions.
PathDwellSec	Time duration for the path
AfterPathSec	Time from end of path to next action.
Temporal/Transition Features	
τ(SearchAct)	Time between two search actions
DayOfWeek	Day of the week
TimeOfDay	One of three 8-hour windows
qq(WordDelta)	Word length change between queries

Table 3: **Events and parameterizations considered in learning predictive models.**

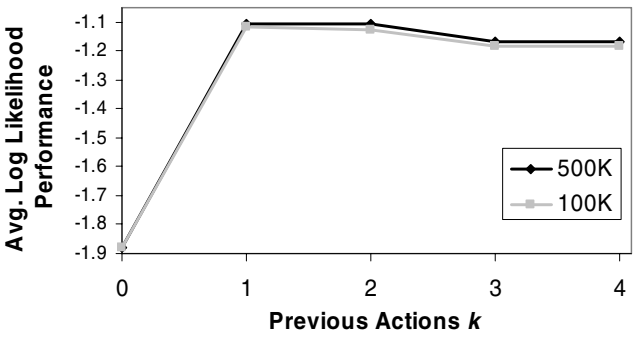

Figure 2: **Predictive performance as event history varies.** Considering search actions beyond the previous one does not increase predictive performance for training sets of 100K and 500K search events.

(z). We distinguish three types of query actions (q-reform, q-same, q-nextpage), capturing the relationship between the next query and the most recently issued query. For the coarse parameterization, b actions that navigate back to the previous result page are treated as occurences of q-same. At the *fine* level of granularity, we consider a total of 13 output values. These values extend the coarse granularity in that q-reform actions are further parameterized by the type of textual modification (*addition*, *deletion*, or *other*) as well as whether or not the user switched search engines. Also, the fine granularity distinguishes clicks on advertisements from other result clicks, and distinguishes back navigation to a previous query from a refresh of the same query. For both granularities, the remaining actions (v, $c_{\neg r}$, and b actions that are not q-same) are not predicted as output values, but are summarized as input variables as in Table 3.

3.3 Locality of Evidence

The SAMlight model considers arbitrarily long event sequences; however, to model the conditional probability $P(a_n|U, S, a_{n-1}, \ldots, a_{n-k})$, a machine learning algorithm requires a finite choice of k. We measured how prediction performance varied with k, the number of previous search actions considered. As can be seen in Figure 2, the performance improves dramatically when considering the previous action, but after $k = 1$ the effect of increasing k is small. In fact, increasing the dimensionality of the parameter space hinders performance slightly, a fact that remains true even when we increase the size of training set five-fold, from 100,000 search events to 500,000. In the remaining experiments, we set $k = 1$ and use the larger training set.

3.4 Prediction Results

In this subsection, we present results of our experiments on predicting users next actions. We predict actions at both levels of granularity (coarse and fine) using a number of elements representing the user, query and session. We also examine the contribution of different elements and temporal dependencies.

The results of the prediction experiments are shown in Table 4. We compare the accuracy of SAMlight with a marginal model, which simply predicts the distribution of next actions observed in the training data (the marginal model's ESS is $q|c_r|b$). Table 4 shows the average log likelihood of each model measured on the test set. Additionally, we list each model's *accuracy*, the fraction of test set actions for which the model's most likely prediction (that is, $\arg\max_{a_n} P_{model}(a_n|U, S, a_{n-1}, \ldots, a_{n-k})$) is correct. SAMlight substantially outperforms the marginal model at both action granularities and for both measures of performance.

	Log Likelihood		Accuracy	
	Fine	Coarse	Fine	Coarse
Marginal	-1.881	-1.497	0.323	0.345
SAMlight	**-1.108**	**-0.859**	**0.615**	**0.643**

Table 4: **Next action prediction.** At both the coarse and fine granularities, SAMlight offers substantial improvements in likelihood and accuracy over a baseline marginal model.

Predictive Value of Different SAM Elements

The SAM contains several different events, each of which can be parameterized. Here, we investigate which of these elements and parameters are most important for predictive accuracy.

We first examine the relative importance of SAM events, and then consider individual features. The fact that the end of session action z is a deterministic function of $\tau(SearchAct)$ (that is, end of session occurs *iff* $\tau(SearchAct) > 30$ minutes) has the potential to overstate the predictive value of temporal delay. To eliminate this trivial dependency, for these experiments we remove from the data all cases in which the next action is z.

Our comparison of different SAM elements is summarized in Table 5. The likelihood performance at both the coarse and fine granularities is shown. The coarse and fine accuracy differences tend to be small for these models (because the event space is dominated by a few highly probable events). Thus, in Table 5 we include accuracy on the specific problem of predicting whether or not the next action is a result click (c_r), where differences between methods are more apparent. Because the test set is large, all non-zero accuracy differences in Table 5 are statistically significant ($p < 0.001$, chi-squared test).

The Previous Action (PA) model uses a single input feature, whether the previous action was q (or, treated equivalently, a back action to R) or c_r, and ignores the other features and SAM elements. PA has an ESS of $(q|c_r|b)(q|c_r|b)$. PA, like SAMlight, is a client-side model. As mentioned in Section 2, many recent search models are server-side models. To quantify the benefits of client-side instrumentation, we compare PA with a model that uses only the previous action observable by the search server (ESS of $(q|c_r)(q|c_r)$). The PA model offers substantially improved performance over both the marginal model and (in terms of likelihood) the server-side model.

We now examine the combination of PA with additional SAM elements. The action features $\{q, c_r, b, v, c_{\neg r}\}$ add the

	Fine	Coarse	c_r Accuracy
Marginal	-1.717	-1.271	0.601
Previous Server Action	-1.398	-1.087	0.688
Previous Action (PA)	-1.284	-0.989	0.688
PA + U	-1.281	-0.985	0.695
PA + S	-1.243	-0.949	0.737
PA + τ(SearchAct)	-1.218	-0.931	0.725
PA + $\{q, c_r, b, v, c_{\neg r}\}$	-1.199	-0.912	0.733
SAMlight	**-1.128**	**-0.840**	**0.772**

Table 5: **Importance of different SAM elements.** The table shows the average log likelihood performance on the coarse and fine prediction tasks, as well as the accuracy of predicting c_r events.

Variable	Effect on $P(c_r)$
τ(SearchAct)	$-$
q(FirstResult)	$-$
q(HasSuggestion)	$-$
S(qFrac)	$-$
q(HasDefinitive)	$+$
q(c_rProb)	$+$
S(Num$q$)	$-$
S(Max$q$Words)	$+$

Table 6: **Most predictive individual features.** Features are listed in order of discriminatory influence, along with an indication of whether each feature varies positively or negatively with the probability that the next action is a result click.

most predictive power in terms of likelihood, while the session features S offer the greatest accuracy improvements. Further, the results show that the predictive power of the SAM elements can be complementary. The SAM elements harnessed in concert (in SAMlight) provide better predictions than models which use the elements in isolation. Somewhat surprisingly, the user variables do not provide substantial improvements over PA. This is probably due to our focus on predicting high-level events, rather than more detailed information (such as individual URLs, where user data has been valuable [Sun *et al.*, 2005; Chevalier *et al.*, 2003]). Lastly, note that the relative likelihood differences between models are similar for both the coarse and finer-grained prediction tasks, suggesting that the two tasks require similar features for effective prediction.

The most predictive individual features are listed in Table 6, along with the direction of their influence on the probability of a result click. We found that the temporal feature τ(SearchAct) is particularly important for improving predictive accuracy. Figure 3 shows the temporal dynamics following a search query in more detail. Many of the features' relationships with result click probability are intuitive—for example, result clicks are less likely for queries with spelling suggestions, but more likely for queries with "definitive" results (these are queries with a predictable destination such as "amazon" or "hotmail"). More interestingly, queries requesting deeper portions of the result set are less likely to result

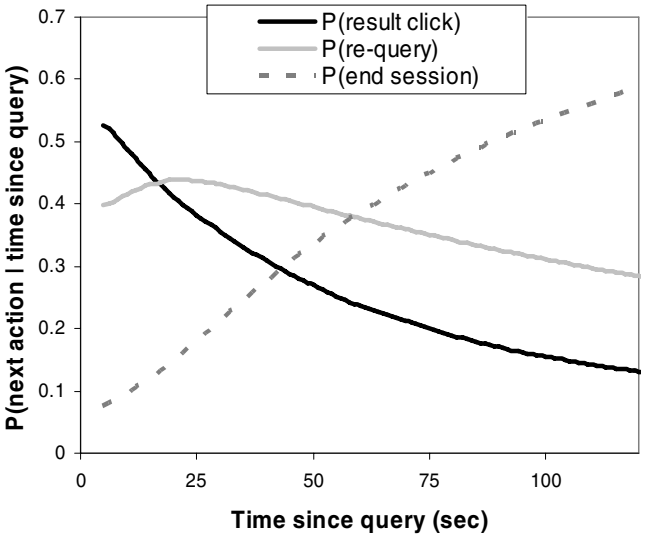

Figure 3: **Effects of delay since previous action.** Probability distribution over the action following a query based on τ(SearchAct), the time since the previous action.

in clicks (i.e., q(FirstResult) varies negatively with $P(c_r)$) and result clicks become less likely as the number of queries S(Num$q$) executed previously in the session increases.

An interesting finding highlighted in Figure 3 is that click probability falls rapidly after a search. In fact, 50% of result clicks occur within 15 seconds after a search. This quick action implies that the latency involved in retrieving a result page contributes substantially to the total duration of a search session. Next, we show how we can decrease this latency by prefetching results the user is likely to access, using spare bandwidth available while the user examines search results.

4 Applications

Studies of user search activity have previously been applied to improve the search experience by aiding with query expansion [Cui *et al.*, 2002], generating reformulation suggestions [Duame and Brill, 2004; Beeferman and Berger, 2000], and improving result ranking [Agichtein *et al.*, 2006]. A rich activity model like SAMlight provides an opportunity for improving the search experience in many ways. Below, we investigate applying the models to the principled prefetching of content.

4.1 Sample Application: Result Prefetching

Prefetching uses idle bandwidth to proactively acquire content a user is likely to access, with the intent of reducing the amount of latency the user experiences. In a prior study of probabilistic policies for prefetching, it was postulated that richer user modeling could enable wiser choices of which content to prefetch, thus providing better bandwidth/latency tradeoffs [Horvitz, 1998]. Here, we test this hypothesis using the search activity models learned in Section 3.

Formally, we assume that the user incurs a *cost* when consuming bandwidth or experiencing latency, and that the costs

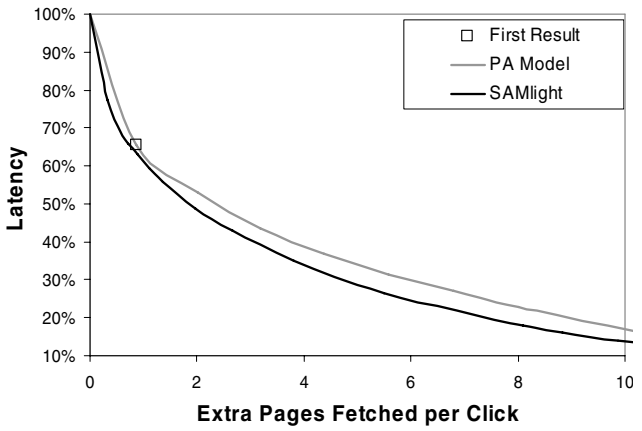

Figure 4: **Prefetching performance.** SAMlight and PA offer a smooth tradeoff between bandwidth (plotted in terms of extra pages fetched per result click) and latency.

of both bandwidth and latency vary linearly with the amount of data accessed. Let α denote the bandwidth cost of loading one page, and let β denote the latency cost of loading the same data. Let p_i be the probability that a particular result page i is requested by the user. The expected increase in bandwidth cost due to prefetching page i is then $\alpha(1 - p_i)$, and the expected reduction in latency cost due to prefetching i is βp_i. To minimize expected cost, we prefetch a result page i *iff* doing so decreases expected cost, that is when $\beta/\alpha > (1 - p_i)/p_i$.

Our models are used to predict the click probability p_i for each result page, and we measure performance as the ratio between α and β varies. We test each of the prefetching algorithms using search and click-through events from the test set, making the simplifying assumption that each page requires four seconds to retrieve, and that pages are prefetched in parallel.

The tradeoff between bandwidth (in terms of extra pages downloaded) and latency is shown in Figure 4. We compare SAMlight and the previous action (PA) model with a baseline algorithm (First Result, indicated with a box in Figure 4) which always fetches the top result (a feature currently available for Google search results in Firefox and Mozilla browsers). The graph shows that substantial latency reductions are possible beyond fetching the first result for both the PA and SAMlight models. The models allow a smooth tradeoff between bandwidth and latency, and using SAMlight provides a somewhat better tradeoff. At a latency reduction of 70%, SAMlight uses 20% less extra bandwidth than the PA model.

The probabilistic models estimate a prefetching policy that minimizes expected cost, given preferences in latency and bandwidth. For different preference values, the probabilistic models can offer cost reductions over algorithms that are not cost-sensitive, like the First Result baseline from Figure 4. In Table 7, we show the cost performance of our models as β/α varies (setting $\alpha + \beta = 1$). We compare against the First Result baseline as well as a second baseline (Prefetch All) which prefetches all results, ignoring costs. The table lists the cost reduction due to prefetching using each method. As preference values vary, cost-sensitive prefetching with PA and SAMlight offers more consistent performance than the baseline algorithms. In particular, PA and SAMlight never increase cost. SAMlight offers the largest cost reductions overall.

	$\beta/\alpha = 1$	5	25	50
First Result	-29%	**13%**	29%	31%
Prefetch All	-576%	-138%	35%	**61%**
PA	**0%**	0%	40%	55%
SAMlight	**0%**	**13%**	**43%**	**61%**

Table 7: **Cost reductions via prefetching.** Shown is the percentage cost reduction (versus the no-prefetch default) for each method, as latency/bandwidth preferences vary. Negative reductions represent increased costs. PA and SAMlight do not increase cost for any preference values, and SAMlight offers the largest reductions overall.

5 Related Work

Several models of user search behavior have been proposed in previous research efforts. We introduced a generalization and formalization of search activities, and focused on modeling the dynamics of search sessions. Section 2 describes in detail the relationship between the general SAM model and several previous models.

Recent work in query analysis (*e.g.,* [Silverstein *et al.*, 1998; Broder, 2002; Beitzel *et al.*, 2004]) has focused on measuring and characterizing the global query distribution (to determine, for example, the average number of words per query, or the proportion of queries having a navigational rather than informational purpose). While we compute global statistics as well, our focus is on modeling the dynamics of search sessions, including sequences of multiple queries along with browsing actions.

In search personalization, models of user interest are constructed by analyzing the content a user retrieves while searching and browsing, along with other data [Shen *et al.*, 2005; Teevan *et al.*, 2005]. These models are then utilized to provide search results tailored to the user's interests. This work is distinct from ours in that it focuses on modeling users, rather than search activity, and is aimed at a specific application. However, the content-focused user models employed in search personalization could be used to augment the behavioral user variables listed in Table 3, potentially improving the predictive accuracy of our models. This is an item of future work.

Lau and Horvitz [1999] developed a dynamic model of search behavior but concentrated exclusively on query-to-query transitions. As in our work, they found that the temporal duration between queries was informative. Our work builds on this by considering additional searching and browsing events, and by providing experiments characterizing predictive accuracy.

Recently, search user interaction data has been applied to improve search engine performance, as discussed in Section

4. Our work is complementary to this work. Instead of focusing on a particular application (such as query expansion), we explore the space of search activity models and the general problem of predicting a user's next action.

6 Summary and Conclusions

We investigated the application of machine learning to logs of Web search activity to build predictive models of users searching for information. We introduced an expressive language for search activity models, and used it to characterize several prior studies. Using a large dataset of search activity, we experimented with building predictive models, and presented analyses showing how different attributes of users, queries, sessions, and temporal delay influence predictive performance. Then, we reviewed several applications of search activity models, and presented an experimental investigation into the use of the predictive models to prefetch content.

We foresee an acceleration of research that couples machine learning with large-scale behavioral data to better understand and support human information-seeking behavior. As part of efforts in this realm, we believe that developing expressive languages for representing, modeling, and communicating about searching and browsing behavior will be valuable for both constructing predictive models and for enhancing ongoing research and collaboration.

Acknowledgments

The first author performed this research during an internship at Microsoft Research and was supported by a Microsoft Research Fellowship sponsored by Microsoft Live Labs. We thank Eugene Agichtein, Andy Edmonds, Max Chickering, and Robert Ragno for providing helpful components and advice for processing and analyzing user activity logs.

References

[Agichtein *et al.*, 2006] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proc. of SIGIR*, 2006.

[Bates, 1989] M. J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5), 1989.

[Beeferman and Berger, 2000] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proc. of KDD*, 2000.

[Beitzel *et al.*, 2004] S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. Hourly analysis of a very large topically categorized web query log. In *Proc. of SIGIR*, 2004.

[Broder, 2002] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2), 2002.

[Chevalier *et al.*, 2003] K. Chevalier, C. Bothorel, and V. Corruble. Discovering rich navigation patterns on a web site. In *Discovery Science*, 2003.

[Chickering, 2002] D. M. Chickering. The winmine toolkit. Technical Report MSR-TR-2002-103, Microsoft, Redmond, WA, 2002.

[Cui *et al.*, 2002] H. Cui, J. Wen, J. Nie, and W. Ma. Probabilistic query expansion using query logs. In *Proc. of WWW*, 2002.

[Davison *et al.*, 2003] B. D. Davison, D. G. Deschenes, and D. B. Lewanda. Finding relevant website queries. In *Proc. of WWW*, 2003.

[Duame and Brill, 2004] H. Duame and E. Brill. Web search intent induction via automatic query reformulation. In *Proc. of HLT*, 2004.

[Fox *et al.*, 2005] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.*, 23(2), 2005.

[Horvitz, 1998] E. Horvitz. Continual computation policies for utility-directed prefetching. In *Proc. of CIKM*, 1998.

[Jones and Fain, 2003] R. Jones and D. C. Fain. Query word deletion prediction. In *Proc. of SIGIR*, 2003.

[Jones *et al.*, 2006] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proc. of WWW*, 2006.

[Lau and Horvitz, 1999] T. Lau and E. Horvitz. Patterns of search: Analyzing and modeling web query refinement. In *Proc. of UM*, 1999.

[Lee *et al.*, 2005] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *Proc. of WWW*, 2005.

[Radlinski and Joachims, 2005] F. Radlinski and T. Joachims. Query chains: Learning to rank from implicit feedback. In *Proc. of KDD*, 2005.

[Rose and Levinson, 2004] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proc. of WWW*, 2004.

[Shen *et al.*, 2005] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *CIKM 2005*, 2005.

[Silverstein *et al.*, 1998] C. Silverstein, M. Henzinger, H. Marais, and M. Moricz. Analysis of a very large altavista query log. Technical Report 1998-014, Digital SRC, 1998.

[Sun *et al.*, 2005] J. Sun, H. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *Proc. of WWW*, 2005.

[Teevan *et al.*, 2005] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of SIGIR*, 2005.