

Predictive Algorithms for Browser Support of Habitual User Activities on the Web

Janez Brank

¹*Jozef Stefan Institute, Ljubljana, Slovenia*
Janez.Brank@ijs.si

Natasa Milic Frayling

Microsoft Research Ltd, Cambridge, United Kingdom
natasamf@microsoft.com

Anthony Frayling

²*i2 Ltd, Fulbourn, Cambridgeshire, United Kingdom*
anthonyf@i2.co.uk

Gavin Smyth

Microsoft Research Ltd, Cambridge, United Kingdom
gavinsmy@microsoft.com

Abstract

Routine user activities on the Web result in the revisitation of Web sites and pages. Standard browser applications provide limited support for this type of habitual behaviour. They typically expose lists of visited URLs that are automatically recorded by the system or manually created by the user, such as bookmarks. Studies have shown that these approaches are not successful in supporting routine user activities. Informed by our user research we designed a browser feature that automatically exposes candidate URLs for revisitation by the user. In this paper we describe and evaluate the algorithms that we use to model the user's habitual behaviour. We demonstrate how a structured navigation history model facilitates the discovery of relevant usage patterns and supports predictive algorithms that are applicable to relatively short personal navigation histories.

1. Introduction

The Web has become an integral part of business practices and social interactions. Consequently, users regularly access the Web to conduct a range of activities. Some of these activities are performed repeatedly over a period of time. Thus, there are sites and pages that users regularly visit, with varying frequency and over different periods of time. However, the standard approach of analyzing the user's navigation history as a simple sequence of URL visits is not suited for identifying patterns of routine activities. Such analyses reveal highly skewed

distributions of revisited URLs, dominated by visits that occur in the short term navigation history. These are mostly due to the Back navigation.

In order to identify revisitation patterns from the user's habitual activities, we need to introduce additional elements in the organization of navigation history. For that we use a history model that derives structure from the user's interaction with the browser as exploited in [10][18]. We identify patterns in the user's activities and proactively presents past URLs in a link bar, predicting those that the user may want to revisit next. Furthermore, the collected navigation data enables us to evaluate alternative algorithms and optimize various aspects of the user navigation. Such explorations and fine tuning are feasible only through simulations and batch mode experimentation. While further user studies are required to confirm that optimized algorithms meet the user's needs, the results of our work significantly contribute to the understanding of the problem area and enable further refinements of the feature design.

In the following sections we provide an overview of the relevant research and describe our approach to analyzing the user navigation history. In Section 3 we describe the algorithms and experimental set up and, in Section 4 we discuss the experimental results. We conclude with a summary of gained insights and plans for future work.

2. Background research

Studies of Web access have shown that the users frequently revisit pages they have seen in the past. Cockburn and McKenzie [5] reported that, on average,

^{1,2} Participated in the research project while working at Microsoft Research Ltd, Cambridge, United Kingdom.

81% of Web accesses are page revisits. A figure of 61% was reported by Catledge and Pitkow [4], and 58% by Tauscher & Greenberg [17]. A large proportion of revisits are due to revisits in the recent navigation history. Tauscher & Greenberg [17] estimate 43% chance that the next user navigation step will be a page from 10 most recently accessed URLs. While users continuously visit new pages [5] [17] they also return to pages from a more distant past. Indeed, about 26% of revisited URLs are not among 10 most recent URLs, not even in the top 20 or 30 [17]. At the moment, the user can access these pages by typing in a URL, with the aid of URL auto-complete features, or by using a bookmark to a page, if one has been created by the user.

URL typing aids are effective when the user can remember the initial sequence of characters and can recognize the correct URL. Bookmarks are used to accomplish various functions, from 'task lists' to archives of 'not to be lost' links. They have not been effective in supporting users' repeated activities. The need for their explicit management by the user and in-time marking also increases the overhead of their usage [1][9].

2.1. Observations and problem definition

User studies in [10][18] show that users' activities on the Web may result in a single or several navigation paths. They motivate a structured model of the navigation history that comprises navigation sessions and web trails.

Navigation Sessions are defined as periods of Web access that are separated from each other by noticeable periods of the browser's inactivity. This definition provides flexibility of separating online navigation from the management of application windows. Indeed, we need not assume that a navigation trail or Web activity is performed within a particular browser window.

WebTrail, as in [10][11][17][18], designates a sequence of navigation steps that comprise link executions and Back and Forward navigations. In our definition, the trail is initiated by typing a URL into the address bar, or by selecting a URL from bookmarks or another document containing a link. Beginnings of trails will therefore be considered as starts of user activities.

Previous analyses of navigation histories are focused on estimating the likelihood of predicting the very next revisited URL [17]. Our objectives are to predict the next recurring user activity by predicting the start of the corresponding trail(s) and related pages. We also aim at algorithms that give sensible predictions from start, when no history is available, and are sufficiently robust to learn from scarce and noisy data since user logging may not be systematic and continuous.

2.2. Predictions on the Web

Discovery of patterns and predictions have been researched in the context of various Web usage areas.

Srivastava et al. [15] offer a detailed overview of research areas, projects, and commercial services that involve Web usage mining.

2.2.1. Application areas. Optimization of Web server performance has become increasingly important with the growing number of users on the Web. Web server logs are mined to identify traffic patterns [12] and develop policies for Web caching, pre-sending, load management, and site redesign. A related application area is a proactive recommendation of URLs by online services. Search engines, shopping sites, and similar online services make dynamic recommendations based on patterns in the behaviour of the community of users. They use aggregate information across users to recommend new links or items to an individual. Finally, there has been work on information agents such as Letizia [8], WebWatcher [7], and SiteHelper [13] which exploit individual users' navigation patterns and contents of viewed pages to create user profiles. These are primarily based on information retrieval techniques for matching user or context profiles with the content of new pages.

Our research focus is different in two main aspects. First, we consider the problem of revisiting sites and pages that the user has seen in the past, as opposed to recommending new content. Second, we work with the evolving and rich record of navigation history of an individual user rather than aggregate information across users.

2.2.2. Methods and algorithms. Existing literature describes clustering methods that are applied to usage logs for classifying users and personalizing recommendations of new content [12]. Mining of association rules, for example, provides a prediction of whether a particular item will be requested by a user if a set of other items have been already accessed within a session. Similarly, there are methods that take into account the order in which items are observed, mining sequential patterns [6][3]. The design and evaluation of these methods assume the existence of large data sets from the server side logging. Accordingly, the research has focused on the computational efficiency [2].

In our case, the available data sets are rather small. Consequently, we are not concerned with computational complexities but with the sparse data problem. We devised new methods that are appropriate for such conditions and compare them with existing methods for mining association rules [2] and navigation sequences [3][14].

3. Browser feature design

Our objective was to design a browser feature that proactively exposes two types of links: (1) URLs that the user may access to start the next Web activity, here referred to as *start-of-trail* URLs, and (2) URLs that the user has visited in the past while performing the current activity. We implemented a prototype SmartFavourites which extends



Figure 1: SmartFavourites LinkBar showing start-of-trail links (large purple stars) and page predictions (pairs of small purple stars), with the thumbnail image of the 2nd link.

the Microsoft Internet Explorer (IE) browser with a link bar, displaying the predicted links (Figure 1). We deployed the prototype and formally evaluated its concepts in an observation user study [18].

3.1. User interface

SmartFavourites captures a comprehensive log of the user's interaction with the Browser, including back navigation, forward link executing, URL typing, form filling, automatic and user driven spawning of IE windows, and similar. In the normal usage, the data is continuously accumulated and used for calculating and presenting predictions in real time. The link bar displays a mixture of start-of-trail links, designated by large purple star icons, and page links related to the currently viewed start-of-trail, marked by a pair of small purple stars (see Figure 1). The number of visible recommendations varies with the size of the browser window. Presented set of links may not always contain the small double star items since no such predictions can be made if the user is performing a new activity. Links that represent starts-of-trails are always present, exposing URLs of the next possible user activity.

Our evaluation of algorithms is consistent with the requirements of the interface design. For the sake of consistency, we assume that the desired set of predictions contains up to 3 links of each type. We also eliminate back navigation links from the predictions, since these are captured by the standard Back button.

3.2. Predictive algorithms

We implemented several algorithms for predicting starts-of-trails and relevant pages within trails. While the

user interface in Figure 1 lends itself to optimizations by mixing various types of predictions, here we focus on evaluating individual algorithms, each with regards to its own specific objectives.

3.2.1. Predicting starts-of-trails. Intuitively it seems reasonable to apply a scoring scheme that gives more weight to start-of-trails that occur more frequently throughout the history, i.e., across navigation sessions, and within individual sessions. Also, we may want to promote those that occur more recently in the long term history.

We thus use a cumulative frequency of the start-of-trails across navigation sessions and automatically penalize the absence of a URL in the intervening sessions. We introduce an exponential time decay factor over navigation sessions that enables us to increase or reduce the bias towards recent trails. We also use a normalized frequency of the start-of-trail occurrences within the session to prevent the dominance of user activities that may have occurred frequently within a single session. The resulting algorithm, [ST-Fr] thus involves the following steps:

Algorithm [ST-Fr]. For each URL that corresponds to a start-of-trail in a navigation session n ,

1. Determine the frequency r_n of the URL within the navigation session n
2. Calculate the normalized frequency score f_n from r_n based on the transformation

$$f_n = 2 \times r_n / (r_n + 1)$$

which maps the raw frequencies r_n onto the interval [1, 2). This transformation is inspired by the frequency normalization approaches in information retrieval.

3. Add the frequency score f_n to the cumulative frequency from the previous session S_{n-1} :

$$S_n = \alpha \times S_{n-1} + f_n$$

where α is a *time decay* or *accumulation factor* with the value in the interval $[0, 1]$. $\alpha = 1$ corresponds to no time decay. Thus, after n navigation sessions, the cumulative frequency S_n is calculated as follows:

$$S_n = \alpha^{n-1} \times f_1 + \alpha^{n-2} \times f_2 + \alpha^{n-3} \times f_3 + \dots + \alpha \times f_{n-1} + f_n.$$

3.2.2. Context sensitive exposure of starts-of-trails. In order to identify *user routines*, exhibited as repeated *sets of activities* across navigation sessions, we implemented algorithm [ST-Co]. [ST-Co] is intended for context sensitive exposure of starts-of-trails: if one of the start-of-trails from the routine is detected, we present the others from the group. The algorithm allows us to incorporate the order of URL access, proximity of trails, and co-occurrence frequency:

Algorithm [ST-Co]. For a given navigation session n , we identify the time sequence of starts-of-trails as they occur in the session.

1. For each pair of URLs (a, b) in the navigation session n , identify the frequency of “ a occurring before b ” ($a \ll b$) within the distance d , i.e., separated by $d-1$ starts-of-trails: $c_{n,d}(a \ll b)$.
2. Specify a weighting scheme to reward the proximity of URLs: define the value of the maximum weight M to be associated with the adjacency of two starts-of-trails (at distance $d=1$). For those with distance d , the weight factor is determined as $w_d = M+1-d$, where $d=1, \dots, M$.
3. Calculate the total co-occurrence score for $a \ll b$ in session n :

$$c_n(a \ll b) = \sum_d c_{n,d}(a \ll b) \times w_d.$$

4. Accumulate the co-occurrence score across sessions, applying the decay factor α :

$$C_n(a \ll b) = \alpha \times C_{n-1}(a \ll b) + c_n(a \ll b), \text{ with } 0 \leq \alpha \leq 1.$$

5. If a is visited during the session $n+1$, look up the ranked list $C_n(a \ll x)$ and select the top scored URLs x .

Essentially, we collect co-occurrence statistics for the trail a and M subsequent trails. We retain information about the order of trails by storing separately the co-occurrence scores for $a \ll x$ and $x \ll a$ and can use them individually or combined in our experiments.

3.2.3. Predicting pages associated with trails. We implemented several algorithms to analyze pages in the trails, with different levels of context sensitivity and information about the navigation structure.

Algorithm [PP-Fr] is analogous to [ST-Fr] described in the previous section, except that it looks at the frequency of pages within trails instead of the frequency of starts-of-trails within sessions. The cumulative score of each URL is updated at the end of each trail and the time decay factor is applied accordingly.

Algorithm [PP-Fr-A]. We note that a particular page may

be associated with very different activities. Therefore, it is important to preserve information about the activity, i.e., start-of-trail within which a URL has been accessed. We maintain a cumulative score of a URL for *each distinct start-of-trail*. This score is increased by the URL frequency only if the URL occurs in the trail with the specific start-of-trail. Once we detect that the user has started an activity we can expose the related pages, ranked highly by [PP-Fr-A].

Algorithm [PP-Co] predicts page revisits in relation to the currently viewed page. We calculate co-occurrence statistics among pages analogously to [ST-Co], except that we look at co-occurrences of pages within trails instead of the co-occurrences of starts-of-trails within sessions. We recommend revisits of the pages from past trails that co-occur with the currently viewed page.

We could also explore page co-occurrences only within trails of the particular start-of-trail, as in [PP-Fr-A], but we expect that the page co-occurrence already enforces the coherence of the user activity and thus we omit this option.

Algorithm [PP-Seq] detects repeated navigation sequences of a given length s . It is motivated by the analysis of user logs from the studies in [10][18], where we applied the pattern detection module algorithm (PDM) [6] to identify the longest repeated subsequences in URL visits. We observed that, among sequences longer than two, 63% involve 3 links. Thus we focus on sequences of fixed size $s = 3$ and apply the following steps:

1. For each URL in a trail, generate and keep count of *navigation sequences* of length s that start with that URL.
2. Accumulate the frequencies across trails, applying the time decay factor over navigation trails.
3. If the user accesses URL a , examine the sequences of length s starting with a . Present URLs that occur most frequently as the final (s^{th}) URL in these sequences.

3.2.4. Alternative algorithms. For comparison, we consider methods for mining association rules [2][3]. The basic algorithm starts by considering the data as a collection of sets, often called “baskets”, containing items. It discovers ‘inference’ rules of the form $A \Rightarrow B$, meaning “if a basket contains all items from the set A , it will probably contain all items from the set B ”. The probability that this is true is called the *confidence of the rule*, and the percentage of baskets that contain both A and B is the *support of the rule*. One can define a minimum confidence and minimum support and discover only rules that exceed these thresholds.

In the context of our problem, we treat each navigation session as a basket of start-of-trail URLs and use the resulting rules to predict future visits to start-of-trails. Let T be the set of start-of-trails that have been visited so far in the current session. We use all the earlier sessions to train a set of rules. For each rule $A \Rightarrow B$, if A is a subset of T we consider URLs from B as possible recommendations to the

user. Each URL is scored by the confidence of the most confident rule that recommended it. We use an analogous approach for predicting pages within the current trail: we consider trails as ‘baskets’ of URLs.

Since individual user logs are relatively small, it is less likely to obtain association rules with a high support. Furthermore, the antecedent side of the rule may be fulfilled for only very few rules. This makes it difficult to obtain reliable predictions.

4. Algorithm evaluation

In our experiments we use navigation logs from 30 individuals, some using the standard IE browser and others using IE enhanced with the SmartFavourites prototype [10][18]. Table 1 presents available statistical information about data logs.

We investigate several parameters, including the impact of the time decay α and the frequency at which we update the statistics used in the algorithms: after each navigation session, end-of-trail, end-of-window-session, etc.. Most importantly, we perform a comparison with a reasonably sophisticated baseline for each of the prediction tasks. All comparative results are checked for statistical significance.

4.1. Evaluation measures

At each point in time, algorithms learn from the navigation history that is available up to that point and each prediction algorithm presents up to 3 links. We are interested in the average recall of revisited links for each of the algorithms individually. Essentially, we check whether the user has revisited a page from the past and if so whether the link was among the top k predicted links.

Specifically, the start-of-trail algorithms, [ST-Fr] and [ST-Co], consider all the trails up to the given point in time. If the next revisited start-of-trail is among predictions, we mark it as a success. We calculate the success rate as a percentage of all revisited starts-of-trails that we predicted and average them it across the users.

Page predictors, [PP-Co] and [PP-Seq] are evaluated slightly differently. Their objective is not necessarily to predict the very next URL visit but to provide shortcuts to pages that may not be accessible from the current page. In the user logs these pages appear further in the navigation trail. Thus, the only practical way to evaluate the algorithms is based on the overlap of predictions with the remainder of the navigation trail.

For the predictions at the visit to URL a in the trail t we check whether any URL from the remainder of the trail is included in the predictions. If so, we give a credit of 1 to the predictor. The total score is averaged over all the predictions for which there was at least one revisit present in the remainder of the trail. We provide the average score for predictors that display $k=1$ and $k=3$ links to the user, respectively.

Table 1. Statistics about some of data sets used in the experiments: User Id and the number of sessions, trails, URLs, URL visits, URLs visited more than once, and days.

User	No. Ses.	No. Tr.	URLs	Visits	Vis>1	Days
1	1208	2480	3795	8933	1416	94
2	952	1461	1792	4374	627	74
3	631	1036	1700	3307	559	40
4	429	1046	1334	3070	391	27
5	372	830	1237	1951	325	29
6	432	760	1083	2074	329	24
7	141	355	1010	1727	215	17
8	313	678	923	1674	281	35
9	169	309	893	1920	265	10
10	219	542	758	1649	251	37

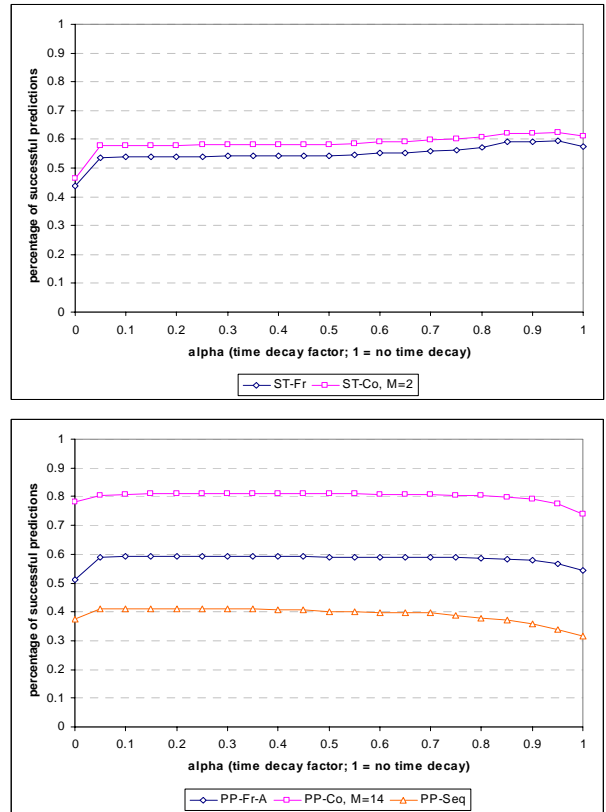


Figure 2. Influence of the time decay factor. The upper chart shows that the time decay parameter does not significantly influence the start-of-trail predictions. The lower chart shows that time decay is important for page predictions.

4.2. Experiment results

4.2.1. Influence of the Time Decay. In the experiments with the start-of-trail prediction, the time decay of

Table 2. Performance of the start-of-trail prediction algorithms: the best performance achieved over different values of the parameters α and M , with the corresponding parameter values in parentheses.

Algorithm	Update weights at the end of each session	Also update weights before the current trail
ST-Fr with raw frequencies (r_n)	0.5247 ($\alpha = 0.95$)	0.5995 ($\alpha = 0.9$)
ST-Fr with normalized frequencies (f_n)	0.5320 ($\alpha = 0.95$)	0.6015 ($\alpha = 0.7$)
ST-Co	0.5233 ($M = 3, \alpha = 0.95$)	0.5888 ($M = 3, \alpha = 0.9$)
Baseline	0.5010	

frequency scores did not produce statistically significant improvement in performance (see Figure 2, upper chart). The best performance was achieved around $\alpha = 0.9$ or $\alpha = 0.95$ (Table 2), but the results are not significantly better than when no time decay is used ($\alpha = 1$).

In the page prediction experiments the time decay parameter is more helpful (see Figure 2, lower chart). The difference between the best performance, achieved for $\alpha < 0.5$, and the performance for $\alpha = 1$ is statistically significant. Equally good results are achieved at a wide range of α with the best values around $\alpha = 0.2$ for both [PP-Co] and [PP-Seq]. We speculate that a more significant impact of α is observed for pages because we are working at a finer granularity of data, i.e., pages rather than trails. Thus, for longer personal histories, we may begin to observe similar effect of time decay for trails.

4.2.2. Prediction of Starts-of-Trails. Table 2 presents results from the experiments with the starts-of-trails algorithms [ST-Fr] and [ST-Co]. For baseline comparison, we initially tried a random selection of 3 URLs from recently visited starts-of-trails. However, by far the best results are achieved when the three most recently starts-of-trails are selected. Thus we use this more effective algorithm for comparison with our algorithms.

In Table 2 we show the highest performance achieved by each method over the range of time decay values, when 3 predictions are made by the system. While it seems that ST-Fr performs slightly better than ST-Co, the difference is not statistically significant. Both methods perform significantly better than the baseline method.

Furthermore, algorithms [ST-Fr] and [ST-Co] use frequency and co-occurrence scores, respectively, which can be updated at the end of the current navigation session or after each completed trail. In the latter case, the scores associated with the trails from the current session are used straight away for predictions. The results in the Table 2

Table 3. Page prediction algorithms: the best average performance when displaying $k = 3$ and $k = 1$ recommended links, respectively. Results for the raw frequency r_n and normalized frequency f_n .

Algorithm	Update weights at the end of each trail	Also update weights before current page
Presenting $k=3$ link recommendations		
PP-Fr, r_n	0.3043 ($\alpha = 0.85$)	0.4704 ($\alpha = 0.05$)
PP-Fr, f_n	0.3181 ($\alpha = 0.85$)	0.4711 ($\alpha = 0.3$)
PP-Fr-A, r_n	0.4740 ($\alpha = 0.3$)	0.5929 ($\alpha = 0.35$)
PP-Fr-A, f_n	0.4768 ($\alpha = 0.45$)	0.5935 ($\alpha = 0.6$)
PP-Co	0.7700 ($M = 18, \alpha = 0.75$)	0.8125 ($M = 14, \alpha = 0.25$)
PP-Seq	0.3041 ($\alpha = 0.9$)	0.4116 ($\alpha = 0.05$)
Baseline	0.4570	
Presenting $k=1$ link recommendation		
PP-Fr, r_n	0.1598 ($\alpha = 0.85$)	0.2475 ($\alpha = 0.1$)
PP-Fr, f_n	0.1742 ($\alpha = 0.9$)	0.2476 ($\alpha = 0.2$)
PP-Fr-A, r_n	0.3387 ($\alpha = 0.5$)	0.3953 ($\alpha = 0.1$)
PP-Fr-A, f_n	0.3528 ($\alpha = 0.7$)	0.3967 ($\alpha = 0.5$)
PP-Co	0.5808 ($M = 11, \alpha = 0.05$)	0.6076 ($M = 12, \alpha = 0.35$)
PP-Seq	0.1849 ($\alpha = 1$)	0.2501 ($\alpha = 0.1$)
Baseline	0.2204	

show that continuous frequency score updates during the navigation session yield a higher performance across the algorithms and the resulting difference is statistically significant. This is not surprising, since our exploration of the baseline algorithm has already indicated that users are likely to repeat recent activities. We also observe that [ST-Fr] does not benefit from the frequency normalization. This is expected because the navigation sessions tend to be relatively short and rarely contain multiple trails with the same start URL.

4.2.3. Prediction of Pages. Table 3 shows results of predicting pages using [PP-Fr], [PP-Fr-A], [PP-Co], and [PP-Seq] algorithms. For comparison, we used a baseline algorithm that recommends the last three visited URLs after the immediately preceding page is removed. We investigated alternative predictors, based on random selections of recent pages, but they performed worse.

The results in Table 3 show the best prediction performance averaged over users and obtained across a range of decay values α . For completeness, we report both the performance when the user is provided with a single ($k=1$) and three ($k=3$) recommendations but focus our discussion on the latter. The maximum performance is

achieved with [PP-Co] and α around 0.2. This implies that the co-occurrence with pages from the previous trails does not play a significant role in supporting users in page revisitation.

We evaluate algorithms [PP-Co] and [PP-Seq] by updating frequency scores of individual pages in two ways: (1) after the entire trail is completed and (2) after each individual page is viewed. More frequent updates produce statistically significant performance improvement, as shown in Table 3. The [PP-Seq] algorithm, i.e., the prediction of pages based on the fixed 3-step navigation sequences alone performs poorly. This is expected as such sequences are rare compared to other types of revisitation.

Finally, we investigated the influence of the parameter M which controls the size of the trail segments that are used for collecting page co-occurrence statistics. For our data sets, using small values of M , $M < 5$, is detrimental to the prediction performance. Similarly, the performance degrades for large values of M , $M > 20$, but much more gradually. If we optimize the value of M for individual users, we observe an increase in the average [PP-Co] performance from 0.8125 to 0.8350.

4.2.4. Predictions using Association Rules. We compare our [ST-Co] and [PP-Co] algorithms with the standard method for discovering and applying association rules (Section 3.2.4). The experiments show that, in over 90% of cases, none of the rules have the antecedent (left-hand) side matched by pages in the current session. Thus, no link predictions can be obtained based on this approach. Even setting the threshold for the required minimum support at only 2 sessions or trails does not improve the performance. Therefore, we conclude that, while association rules may be helpful for analyzing large-scale web usage data, the logs of individual users do not contain enough repetitive patterns to yield useful rules.

5. Summary

In this paper we address the need to support users in routine Web activities and demonstrate how a structured history model can be used to analyze usage patterns and pro-actively present links for Web revisitation. We design and evaluate algorithms which take into account statistics about Web trails and constituent pages. We demonstrate that these algorithms outperform simpler approaches that do not utilize the structure. We also demonstrate that the standard association rules method does not produce useful predictions on the individual user's logs that we are working with.

One natural extension of the presented work is to introduce more detailed typing of page visits and use them in the prediction algorithms. Furthermore, it is important to investigate how the presence and usage of recommendations themselves affect the user behaviour and subsequent predictions of the system. Finally, combining

Web trail and page level predictions in the display presents further opportunities for optimizing the usefulness of the SmartFavourites and similar features.

6. References

- [1] Abrams, D., Baecker, R. and Chignell, M. Information Archiving with Bookmarks: Personal Web Space Construction and Organization. In Proc. of CHI'98 (1998), 41-48.
- [2] Agrawal, R. & Srikant, R. Fast Algorithms for Mining Association Rules. In Proc. of 20th Int. Conf. on VLDB (1994).
- [3] Agrawal, R. and Srikant, R. Mining Sequential Patterns. In Proc. of the 11th Int. Conference on Data Engineering (1995).
- [4] Catledge, L., and Pitkow, J. Characterizing browsing strategies in the World Wide Web. Computer Networks and ISDN Systems 27(6) (1995), 1065-1073.
- [5] Cockburn, A. and McKenzie, B. What Do Web Users Do? An Empirical Analysis of Web Use. Int. J. of Human-Computer Studies 54 (2001), 903-922.
- [6] Craw, D. & Smith, B. DB_Habits: comparing minimal knowledge and knowledge-based approaches to pattern recognition in the domain of user-computer interactions. In R. Beale & J. Finlay, Eds. Neural Newt. and Pattern Recognition in Human-Computer Interaction. Ellis Horwood (1992), 33-61.
- [7] Joachims, T., Freitag, D., and Mitchell, T. WebWatcher: A tour guide for the WWW. In Proc. of the 15th Int. Conf. AI (1997).
- [8] Lieberman, H. Letizia: An Agent that Assists Web Browsing. In Proc. of the Int. Joint Conf. on Artificial Intelligence (1995).
- [9] Maarek, Y. and Ben Shaul, I. Automatically Organizing Bookmarks per Contents. In Proc. of the 5th Int. WWW Conf. (1996).
- [10] Milic-Frayling, N., Jones, R., Rodden, K., Smyth, G., Blackwell, A. and Sommerer, R. SmartBack: Supporting Users in Back Navigation. In Proc. of the 13th WWW Conf. (2004).
- [11] Milic-Frayling, N., Sommerer, R., and Rodden, K. WebScout: Support for revisitation of Web pages within the navigation session. In the Proc. of IEEE/WIC Int. Conf. on Web Intelligence (2003), 689-693.
- [12] Mobasher, B. & Cooley, R. Automatic Personalization Based on Web Usage Mining, In Comm. of ACM 43(8) (2000), 142-151.
- [13] Ngu, D.S.W. & Wu, X. SiteHelper: A Localized Agent that Helps Incremental Exploration of the World Wide Web. In Proc. of the 6th International WWW Conf. (1997).
- [14] Pitkow, J. and Pirolli, P. Mining Longest Repeating Subsequences to Predict WWW Surfing. In Proc. of USITS'99.
- [15] Srivastava, J., Cooley, R., Deshpande, M., and Tan, P-N. Web Usage Mining: Discovery and Application of Usage Patterns from Web Data. In SIGKDD Explorations (2000).
- [16] Srikant, R. & Agrawal, R. Mining Generalized Association Rules. In Proc. of the 21st International Conference on Very Large Data Bases, Zurich, Switzerland (1995).
- [17] Tauscher, L. & Greenberg, S. How people revisit web pages: empirical findings and implications for the design of history systems. Int.J. of Hum. Computer Studies 47(1) (1997), 97-137.
- [18] Milic-Frayling, N., Jones, R., Rodden, K., Smyth, G. and Frayling, A. Designing for web revisitation: exploring structure from user interaction and navigation. Microsoft Research Technical Report MSR-TR-2004-97 (2004).