

Predicting Web Information Content

Tingshao Zhu, Russ Greiner

Dept. of Computing Science, University of Alberta
Canada T6G 2E1, {tszhu, greiner}@cs.ualberta.ca

Gerald Häubl

School of Business, University of Alberta
Canada T6G 2R6, Gerald.Haeubl@ualberta.ca

Abstract

In this paper, we propose a novel method to infer the web user’s Information Content (*IC*), which is the information that the user must examine to complete her task. In particular, our method learns to predict which *words* (called IC-words) will be in these essential web pages (IC-pages). We first collected relevant training data using an empirical study, where users explicitly identified which pages were IC-pages. We then examined page-content information from these clickstreams, to determine “browsing properties” of each individual word w — *i.e.*, how often was w in the title of a page in each session, or in the anchor to a page that was followed, or a link that was skipped, etc. This training data also labeled each word as an IC-word or not. We used this to train a classifier to identify the browsing properties associated with IC-words. Notice this classifier can predict which words are *IC* given *any* page sequence, even if those pages are in web-sites that have not been visited previously. Our empirical results indicate that this method can predict web users’ *IC* fairly accurately.

1 Introduction

While the World Wide Web contains a vast amount of information, it is often difficult for web users to find the particular pieces of information that they are seeking. This is why some web-sites provide personalized service, which typically observes a user’s navigation through a sequence of pages, and then suggests pages that may provide relevant information (*cf.*, [Mobasher *et al.*, 1999; Perkowski and Etzioni, 1997]). Most such systems are based on correlations amongst the pages that various users visit. Unfortunately, there is no reason to believe that these correlated pages contain useful information — indeed, they may correspond simply to the paths that others have taken towards their goals, but not the goal page itself, or worse, simply to standard dead-ends that users seem to hit.

Suppose one user browses an on-line computer store to buy monitor, keyboard, and mouse, etc. for her¹ new computer.

¹We will use the female pronouns (“she” and “her”) when refer-

The page sequence is shown as the upper part in Fig. 1. The shopcart icon identifies the pages where she makes the purchases.

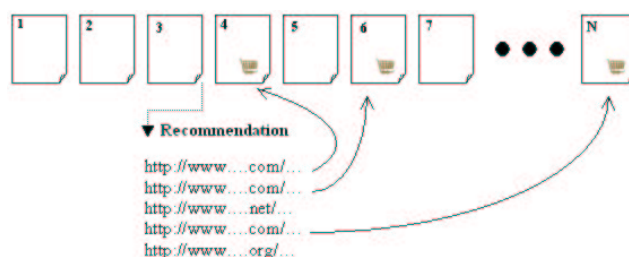


Figure 1: Recommendations for Computer Store Browsing

Notice that the user can complete her task by only visiting these shopcart-ed pages, which therefore correspond to IC-pages. Unfortunately, she had to waste a great deal of time reading through all the other pages, to reach these important ones.

The goal of our research is a recommendation system, ICPF, which can point the users to *informative* pages, *anywhere* in the Web. Like most recommendation systems, our ICPF watches a user as she navigates through a sequence of pages, and suggests pages that (it hopes) will provide the relevant information. For example in Fig. 1, after ICPF has seen the user visit the first three pages, it may produce a list of web pages as recommendations, such as the one shown as the lower part in Fig. 1. If successful, many of these recommended pages would correspond to the actual IC-pages — here the 4th, 6th and n-th page.

ICPF differs in several respects from other recommendation systems. First, our system explicitly attempts to recommend *IC-pages*. By contrast, many other recommendation systems determine other users that appear similar to the current user U , then recommend that U visit the pages that these purportedly-similar users have visited. Unfortunately, there is no reason to believe that these correlated pages will contain information useful to U . Second, as many recommendation systems are server-side, they can only provide information about one specific web-site. By contrast, our client-side

ring to users, of either gender.

ICPF is not specific to a single web-site, but can point users to pages *anywhere* in the Web. The fact that our recommendations cover the entire Web leads to a third problem: *support*. As any single web-site has a relatively small number of pages, a web-site-specific recommendation system can expect many pages to have a large number of hits; it can therefore focus only on these high support (“highly-visited”) pages. Over the entire WWW, however, very few pages will have high support. Our system must therefore use a different approach to find recommended pages.

To learn, and later evaluate, our recommendation system, we collected a set of *annotated web logs*; a sequence of web pages that a user visits, where the user has explicitly labelled each page with a bit that indicates whether this page is “*IC*” — *i.e.*, essential to achieve the user’s specific goal.

Collectively, the 129 participants in the study requested 15,105 pages, and labeled 1,887 pages as *IC*, which corresponds to 14.63 *IC*-pages per participant. This involved 5,995 distinct URLs, meaning each URL is requested 2.52 times on average. Of these, 3,039 pages were search pages (from 11 different search engines); if we ignore these, we find each non-search-engine page was visited only 2.02 times on average. Table 1 shows how often each page was visited; notice 82.39% of the URLs were visited only one or two times. Clearly very few URLs had strong support in this dataset; this would make it very difficult to build a recommendation system based on only correlations across users in terms of the pages they visit.

Table 1: Number of Requests vs Percentage of the URLs

Number of Request(s)	Percentage of the URLs
1	58.93%
2	23.46%
3	7.63%
4	4.08%
5	1.85%
6	1.16%
7	0.88%
8	0.40%
9	0.40%
10	0.18%
...	...

As anticipated, our collected data show that the users of our client-side system did in fact examine pages from many web-sites, throughout the web. It also shows low support for each individual web page, which makes it difficult to build recommendation system based on only correlation. We therefore need a different approach.

We break the task of *IC*-pages prediction down into two stages: (1) predicting what information she wants, which corresponds to the *IC*-words, then (2) locating and returning pages that provide this information (the *IC*-pages). There are several ways to accomplish the second task: We can launch a web agent to “scout ahead” — following links from the current page, seeking a page that matches the user’s *IC*. Alternatively, we can send an appropriate query to a search engine,

based on the *IC*-word, then return/scout those responses.

This report focuses on the first task. We describe a system that learns a classifier to predict which words will be in *IC*-page. The features for each word w are its browsing behavior — *e.g.*, of the times when w appears in the anchor, how often does follow that anchor to a webpage? Or when w appears in the title of some page, how often does she go forward from that page, versus going back to the previous page. Etc etc etc. We train our classifier based on these browsing properties to characterize the kind of words will be in *IC*-page.

Section 2 contrasts our approach with some related works. Section 3 shows how we used the collected information to learn this “*IC*-word” classifier, and in Sections 4 and 5 present more details about data preprocessing. Finally, Sections 6 and 7 report the testing results of our approach.

2 Related Work

Many groups have built various types of systems that recommend pages to web users. This section will summarize several of those systems, and discuss how they differ from our approach.

Our system is seeking the web pages that provide the information that the user wants — *i.e.*, that satisfy the user’s Information Need. Chi et al. [Chi *et al.*, 2001] construct Information Need from the context of the hyperlinks that the user followed, and view it as the information that the user wants; this appears very similar to our approach. However sometimes the context of the hyperlink gives only some hints for the destination information, but not useful information (*e.g.*, “click here”). Moreover, there is no reason to believe that the context around the followed hyperlink is sufficient to convey the user’s intention; notice this does not consider the hyperlinks that are skipped, nor when the user backed up, etc. By contrast, our approach assumes that some browsing properties of a word (*e.g.*, context around hyperlinks, or word appearing in titles or ...) indicate whether a user considers a word to be important; moreover, our system has *learned* this from training data, rather than just assume it will be some specific field.

Billsus and Pazzani [Billsus and Pazzani, 1999] trained a NaïveBayes classifier [Duda and Hart, 1973] to recommend news stories to a user, using a Boolean feature vector representation of the candidate articles, where each feature indicates the presence or absence of a word in the article. That model used a set of words that were *hand-selected* to ensure that they covered all the topics in these articles. Their research, however, provided no explicit feedback from the subject; instead they only inferred the interestingness of the news story from the listener’s actions, such as channel changes. Moreover, their use of hand-selected words poses two problems: First, it places a burden on the user (or system developer) to provide these words. Second, it is difficult to guarantee that the selected words can cover all possible articles — *i.e.*, it is not clear the trained model would be able to make predictions if the user began visiting a completely different set of WWW pages or news stories.

Jennings and Higuchi [Jennings and Higuchi, 1993] trained one neural network for each user to represent a user’s prefer-

ences for news articles. For each user, the neural network’s nodes represent words that appear in several articles liked by the user and the edges represent the strength of association between words that appear in the same article.

Anderson and Horvitz [Anderson and Horvitz, 2002] built a NaïveBayes to predict the candidates (pages or topics) that the user will view next in the session, where the candidates are selected from the previous pages or topics in the same session. By contrast, our ICPFs expected to identify the IC-page, not the likely request pages. Moreover, our system does not limit the recommendations to be only from the current session.

We also view *IC* prediction as a classification task, but instead of building our model based on some pre-specified words, our model is based on “browsing features” of the words, such as how many times the word is in the hyperlink’s anchor text, how many times the word is in the search keyword list, etc.; see Section 5. After training, our system may find some patterns like “any word appears in the three consecutive pages will be in the IC-page”. Note this is different from systems that produce association rules [Agrawal and Srikant, 1994] — *e.g.*,

If a user visits page H,
then she will also examine page J.

as those association rules can only predict pages that have been seen; note we can apply our system to make predictions about pages that have not been visited.

Our approach also differs from systems that involve a set of predefined words — *e.g.*,

After observing a sequence U
if “web” $\in W(U)$ with weight 0.9, and
“software” $\in W(U)$ with weight 0.8, ...,
then page T may be interesting.

We are not looking for patterns based on specific words nor only for specific users, but rather for more general patterns across different sessions and different people, which we expect to be useful even in a new web environment.

Our research identifies “Information Need” with the distribution over IC-words, which relates to the words that will be in the IC-page. Some other systems define the user’s information need as a learned combination of a set of (possibly pre-defined) words — *e.g.*, a NaïveBayes model that classifies each webpage as *IC* or not, using a set of words as the features [Billsus and Pazzani, 1999]. Our method differs as we do not limit the set of words, but instead label each individual word in the user’s current session pages with a measure of its likelihood of appearing within an IC-page. We can use this information to score any given page (*e.g.*, the ones found by our scout) based on the number of high-scored words it contained, or we could form a query to a search engine as a list of the highly-scored words.

3 Learning Task

This section shows how we use the annotated weblogs to build a classifier for characterizing which *words* will appear in the IC-page. As motivated above, we are seeking general patterns that describe how the user locates useful information.

For reasons described above (see also Section 2), these patterns are not based on a specific set of pre-defined words, but rather on the user’s observable behavior in response to the information within the pages visited — *i.e.*, how the words contained in these pages influence her navigation behavior.

We therefore collect this type of information about the words — how they appeared on each page, and how the user reacted. This is based on our assumption that there are general models of goal-directed information search on the web — some very general rules that describe how users locate the information they are seeking. If we can detect such patterns, and use them to predict a web user’s current information need, we may provide useful content recommendations.

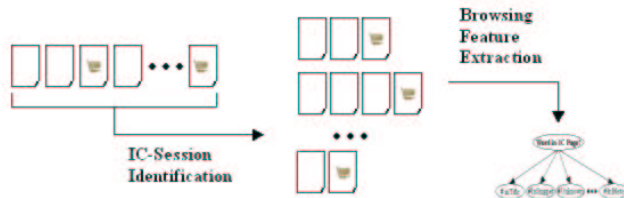


Figure 2: Information Content (*IC*) Prediction

Fig. 2 presents the whole process of our *IC* prediction. The first step (Section 4.1) is to segment each user’s complete clickstream into a set of so-called *IC-sessions* — each of which is a sequence of pages that ends with an IC-page. Section 4 discusses some techniques we use to clean this data. Within each IC-session, we extract all the words in the preliminary non-IC-pages, then collect various “browsing features” of each word. By examining the associated IC-page, we also label each such word as an IC-word or not. Section 6 shows how our ICPF uses this information to train a classifier to predict when a word will be an IC-word.

Note that the performance system does NOT require that the user annotate the webpages; this is just done in the training phase. However, if the user is willing to do this labelling, we could hone the system to the nuances of the current user, and anticipate being able to obtain superior results (compared to a generic system, based on only the base population of users.)

4 Data Preprocessing

To convert the raw log data to be suitable for learning, several steps must be done for preprocessing. Some preprocessing techniques here differ from those introduced in [Cooley *et al.*, 1999].

4.1 IC-session Identification

Each user will be pursuing several different information needs as she is browsing. To identify and distinguish these needs, we must first separate the pages into a sequence of “IC-sessions”, where each such IC-session pertains to a single information need. In general, each IC-session is a consecutive sequence of pages that ends with an IC-page, or the end of the user’s entire session.

Algorithm ICSI:(URLsequence $U = \langle u_1, u_2, \dots, u_n \rangle$):
 outputs Sequence of IC-sessions
 F : Boolean; % true iff current page is immediately after an IC-page
 L : Queue; % stores the current session
BEGIN
 Set $L :=$ empty queue; $F :=$ false
 For $i=1..n$ do
 If u_i is an IC-page then
 If L is not empty, Output L
 $F :=$ true;
 Else
 If (F) then
 If u_i is a search query page then Empty(L);
 If u_i is in L then Pop off L every page after this first u_i ;
 $F :=$ false
 Append u_i to L
 If L is not empty, Output L .
END

Figure 3: ICSI Algorithm

Chen et al. [Chen *et al.*, 1996; 1998] terminated each session on reaching a Maximum Forward Reference (MFR) — *i.e.*, when the user does not follow any outlinks from a page. Of course, these final MRF pages need not correspond to IC-pages. Cooley et al. [Cooley *et al.*, 1999] used time-outs to identify sessions: if the time between consecutive page requests is greater than a threshold, they assume that a new session has started. While the fact that a user remained at a single page may suggest that that page could be *IC*, there could also be other reasons. Note that neither set of authors claims that these final pages addressed the user’s information need, and so they provide no evidence that these pages were IC-pages.

In our case, since we focus on goal-directed browsing, we terminate a session on reaching an IC-page. However, it is not clear that the next session should begin on the subsequent page. For example, imagine reaching an index page I after visiting a sequence of pages $A \rightarrow B \rightarrow C \rightarrow I$, and moreover, I contains a number of useful links, say $I \rightarrow P_1$ and $I \rightarrow P_2$, where both P_1 and P_2 are *ICs*. Here, each IC-session should contain the sequence before the index page since they also contribute to locating each of the IC-pages — *i.e.*, given the browsing sequence $A \rightarrow B \rightarrow C \rightarrow I \rightarrow P_1 \rightarrow I \rightarrow P_2$, we would produce the two IC-sessions $A \rightarrow B \rightarrow C \rightarrow I \rightarrow P_1$ and $A \rightarrow B \rightarrow C \rightarrow I \rightarrow P_2$.

To identify meaningful IC-sessions, we used the heuristic that if the page after an IC-page is a new search query, then a new session starts, since it is very common that when one task is done, users will go to a search engine to begin the next task. Figure 3 summarizes our IC-session identification algorithm.

4.2 Data Cleaning

Our system parses the log files to produce the sequence of pages that have been downloaded. Unfortunately some of these pages are just advertisements, as many web pages will launch a pop-up ad window when they are loaded. As few of these advertisement pages will contribute to the subject’s information needs, leaving them in the training data might

confuse the learner. We therefore assembled a list of advertisement domain names, such as: `ads.orbitz.com`, `ads.realcities.com`, etc. We compare each URL’s domain name with the ad server list and ignore a URL if it is in the list.

We defined each IC-session as composed of *pageviews*, where a pageview is what the user actually sees. In the case of frames, a pageview can be composed of a number of individual URLs. When a frame page is being loaded, all of its child pages will be requested by the browser automatically; and thus instead of recording only the frame page in the log file, all of its child pages will be recorded too. This is problematic when the participant browses within a frame page.

Finally, while we did record the time information, we were unable to use it in the learning process. This is because many subjects switched modes (to “Report mode”) on finding each IC-page, which means that much of the time between requesting an IC-page and the next page was not purely viewing time, but also includes the time spent writing this part of the report. Unfortunately, we did not anticipate this behavior, and so we did not record the time spent in *Report* mode.

5 Attribute Extraction

We consider all words that appear in all pages, removing stop words and stemming, using standard algorithms [Porter, 1980]. We then compute the following 25 attributes for each word w_i , from each IC-session: (In all cases, if the URL refers to a frame page, we calculate all the following measures based on the page view.) Note that each feature is with respect to a single IC-session. Our companion paper [Zhu *et al.*, 2003] gives more detail of the attributes.

Search Query Category

As our data set includes many requests to search engines, we include several attributes to relate to the words in the search result pages.

Each search engine will generate a list of results according to the query, but the content of each result may differ for different search engines. We consider only information produced by *every* search engine: *viz.*, the title (*i.e.*, the first line of the result) and the snippet (*i.e.*, the text below the title). For example, in Figure 4, the title of the first result is “Workshop Home Page”, and its snippet is “Workshop on Web Mining April 7, 2001 on all aspects of Web mining. . .”.

We tag each title-snippet pair in each search result page as one of: *Skipped*, *Chosen*, and *Untouched*. If the user follows a link, the words in its title and snippet will be considered “*Chosen*”. The words that appear around the links that the user did not follow, before the last chosen one, will be deemed “*Skipped*”, and all results after the last chosen link in the list will be “*Untouched*”. Figure 4 shows 2 “*Skipped*”, 2 “*Chosen*”, and 1 “*Untouched*” results. Notice this corresponds to several visits to this search result page: The second entry “Web Mining” is the first one followed; the user later clicks back to the search page, and chooses the fourth entry, “The Data Mining Group”. Also, for pages in general, we say a hyperlink (in page U) is *backed* if the user followed that link to another page, but went back to page U later. A page is

Workshop Home Page
 Workshop on **Web Mining** April 7, 2001
 on all aspects of **Web mining**. ...
www.lans.ece.utexas.edu/workshop_index.htm

Web Mining
 ... **Web mining**, data **mining** terms,
 of interests - clustering (finding natural
www.cs.umbc.edu/~ajoshi/web-mine/

About **Web Design**
 ... Search in this topic. with Jean Kaiser ...
 2 months free! **Web** site hosting ...
webdesign.miningco.com/ - 38k -

The Data **Mining** Group
 Welcome to The Data **Mining** Group **Web** ...
www.dmg.org/ - 5k -

WEBKDD 2002
 WEBKDD 2002. **Web Mining** for Usage
 July 23, 2002, Edmonton, Alberta, Canada. ...
db.cs.ualberta.ca/webkdd02/ - 6k -

Skipped Title
 Skipped Snippet
 Chosen Title
 Chosen Snippet
 Skipped Title
 Skipped Snippet
 Chosen Title
 Chosen Snippet
 Untouched Title
 Untouched Snippet

Figure 4: Title-Snippet State in the Search Result Page

backward if that page has been visited before; otherwise we say a page is forward.

5.1 Sequential Attributes

All the sequential features are extracted from pages in an IC-session except the search result pages and the last IC-page. We also compute the “weight” of each word w , in each page, as $weight(w) = \sum_j N_j(w) \times v_j$, where $N_j(w)$ denotes the number of occurrences of w in the j^{th} “HTML context” [W3C,]: and v_j is the predefined weight associated with this context

For each word w in an IC-session, we compute each of these attributes, and also indicate whether w appears in the IC-page or not. We summarize the browsing properties of all the words along the entire IC-session, with the goal of anticipating what the user is seeking. (Hence, this differs from simply summarizing a single page [Zajicek and Powell, 1997].) Note that when we train the classifier, we do not use the words themselves, but instead just these attribute values and whether the word appears in the IC-page.

6 Classifier Training/Testing

After preparing the data, ICPF trains a NaïveBayes (NB) classifiers on these browsing feature vectors. Recall that NB is a simple belief net structure which assumes that the attributes are independent of one another, conditioned on the class label [Duda and Hart, 1973]. NB also runs fast, and often acquires the performance comparable to other classifiers, such as decision trees and Support Vector Machines. To deal with continuous attributes, we use estimation [Fayyad and Irani, 1993] instead of discretization [John and Langley, 1995], as we found the former works better.

According to the labelled log data, only 12.5% of the pages were IC. Moreover, the number of non-IC-words is

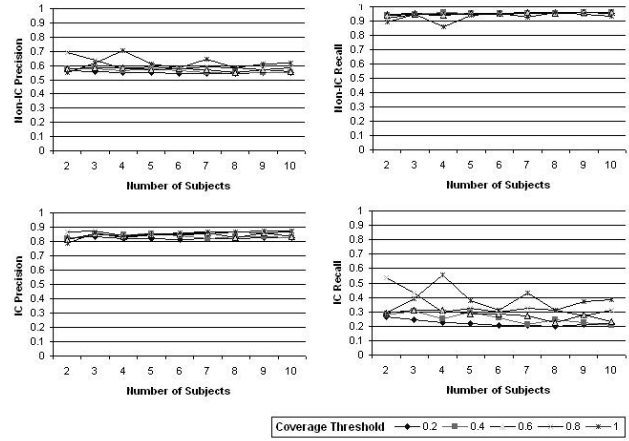


Figure 5: Non-IC and IC-word Prediction Results

far greater than that of IC-words. To deal with this imbalanced dataset, we have tried both down-sampling [Ling and Li, 1998] and over-sampling [Japkowicz, 2000], and found that down-sampling produced more accurate classifiers than over-sampling. (E.g., it produced about 20% higher recall of IC-word prediction than over-sampling.) We then randomly selected an equal number of positive (IC-word) and negative (Non-IC-word) instances as testing data, then generated our training data from the remaining data by randomly removing negative instances until obtaining a number equal to the number of positive instances.

For each IC-session $U = \langle u_1, u_2, u_3, \dots, u_N \rangle$, where u_N is the only IC-page, we let $U_{N-1} = \langle u_1, u_2, \dots, u_{N-1} \rangle$ be the preliminary non-IC-pages. In general, let $W(U)$ be all the words in the set of pages U .

For each $R = 2, 3, \dots, 10$, we randomly selected 20 different groups of size R from the set of participants. Note that we allowed overlap among these 20 groups. For now, we restrict our attention to only those sessions with $W(u_N) \subseteq W(U_{N-1})$ — i.e., where all words appeared in the IC-pages occur somewhere in U_{N-1} .

For each of these 9×20 groups, we call the recommendation function on U_{N-1} to generate the word set predicted as IC-words, which we denote as $WP(U_{N-1})$. (These are the words $w \in W(U_{N-1})$ whose posterior probability of being an IC-word is greater than 0.5.)

We computed four quantities for each group — precision and recall, for both IC-words and non-IC-words:

$$\begin{aligned}
 \text{ICprecision}(U) &= \frac{|WP(U_{N-1}) \cap W(u_N)|}{|WP(u_N)|} \\
 \text{ICrecall}(U) &= \frac{|WP(U_{N-1}) \cap W(u_N)|}{|W(u_N)|} \\
 \text{nonICprecision}(U) &= \frac{|WP(U_{N-1}) \cap \overline{W(U_{N-1})}|}{|WP(U_{N-1})|} \\
 \text{nonICrecall}(U) &= \frac{|WP(U_{N-1}) \cap \overline{W(U_{N-1})}|}{|\overline{W(U_{N-1})}|}
 \end{aligned} \tag{1}$$

where $\overline{W(\cdot)}$ is the obvious complement.

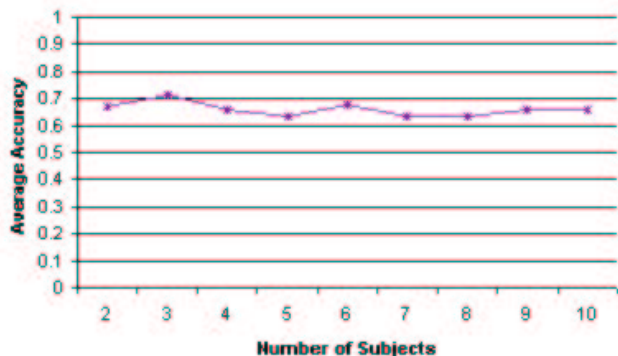


Figure 6: Accuracy of IC Prediction

We built 10-fold training/testing datasets for each of the 9×20 groups, and computed the median value of these 9 results for each of the 4 quantities. (We used the *median* as it is less sensitive to outliers than the mean.)

Here, we found an average accuracy of around 65–70% (Fig. 6); the precision and recall results appear in Figure 5. Even though the average recall of IC-words is about 45%, we anticipate this will be good enough to find IC-pages, which of course is our ultimate goal. Section 1 presented two methods that ICPF can use to locate IC-pages given IC-words. (1) ICPF can scout ahead to find the IC-pages that match the predicted IC-words; knowing 45% of the words on that page makes it easy for the scout to correctly identify the page based on its content, or at least some pages very similar to it. Alternatively, (2) ICPF might try to build search queries from the predicted IC-words. Given the high precision of our IC-word prediction, even with recall around 45%, we can anticipate finding tens of words which will surely be in the IC-page. Since the predicted IC-words are exclusive of stop words, they will be quite relevant to the IC-page’s content. We therefore suspect that a query with these relevant words will help retrieve the relevant IC-page. (We are currently exploring these, and other ways to find IC-pages from IC-words.)

7 Leave-One-Out Testing

In Section 6, the testing is based on word level, that is, predicting which word will be an IC-word. When we try to predict actual IC-pages, we are more concerned with the efficiency of the predictor, when applied on a new user. For example, after we have collected some annotated web logs and used them to build a classifier for predicting IC-words, it is important to know whether the predictor will work effectively for a new user. In particular, we first built a classifier using the web logs from one set of users, then tested this classifier on data from a new user.

We identify each user U with 20 randomly selected groups, where each group contains 10 non- U users. Within each group, we used all 10 users’ data as training data. (Here we again use down-sampling to deal with the imbalanced sets.) We learn the NaïveBayes on this data, then test U on that model. We compute the mean and standard deviation over

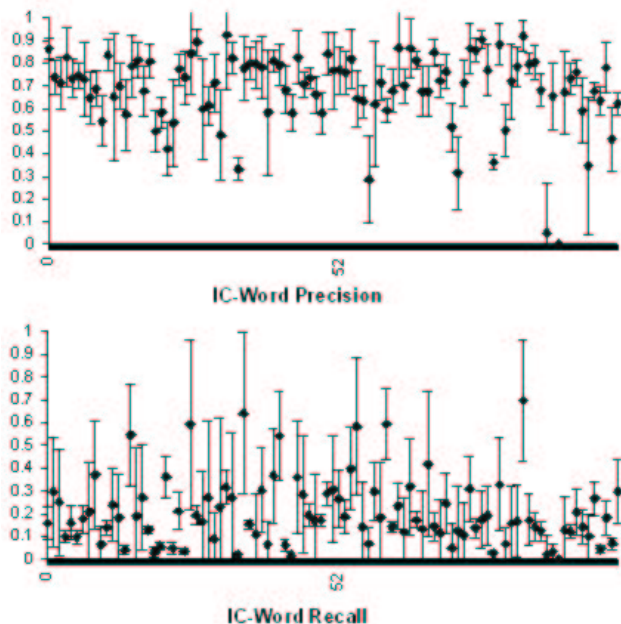


Figure 7: The Average Precision/Recall of IC Prediction for Each Single User

each U ’s 20 groups testing results. Fig. 7 shows the precision and recall of IC-word prediction for each single user.

From Fig. 7, we can see that even though the average value of recall decreased, for each user, there are some groups that can predict U ’s IC-word] family well. This makes us believe that the NaïveBayes learnt based on browsing behavior can be very effective, even for a new user. There may exist a general model for all web users, but it is not so effective when applied on each individual. We believe that when the model is based on a small user group, it can work very well, just as these special groups with high precision and recall for each user. This also suggests that the browsing model from a user group will be much effective for its members, which can provide efficient service even on a new web environment.

8 Conclusion and Future Work

Many recommendation systems apply to only a single website, and basically tell the current user where other similar users have visited. Our goal is a complete-web recommendation system that can point a user to the webpages that actually contain the information that this specific user will need to accomplish her current task, wherever those pages appears, anywhere on the web. To accomplish this, our ICPF first learns a model of general web users. It does this by first extracting properties (“browsing features”) of the words that appear in a training set of the user’s annotated web logs. Our ICPF learns which combinations of these browsing features determine whether the associated word is likely to be included in the IC-page. As our system depends only on *characteristics* of words, and not on the specific words themselves, it can be used to classify the completely different set of words associated with a completely different page sequence, and can

recommend pages from anywhere on the Web.

To assess the usefulness of this system, we conducted a laboratory study to collect realistic annotated data. In this study, subjects perform a series of information-search tasks on the web, and indicate which of the pages they view are IC-pages. Our ICPF then determines under what circumstances a word will be in the IC-page. Our empirical results show that our system works effectively.

Future Work: Most of the results reported here are based on a study involving a travel-planning task. We are planning further studies, to test the generality of our approach in other contexts — *e.g.*, researching academic information or applying to graduate school.

We are currently investigating more effective ways to predict IC-words, and hence IC-pages, perhaps based on yet other features of the IC-session, such as other page content information, or perhaps timing information, etc. We are also exploring the best way to connect our ICPF with a scouting system and/or multiple search engines, and perhaps yet other ways to provide specific page recommendations to the user.

We plan to explore Natural Language processing systems to extend the range of our IC-words, and other machine learning algorithms to make better predictions, and help us to cope better with our imbalanced dataset.

References

- [Agrawal and Srikant, 1994] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th Int'l Conference on Very Large Databases (VLDB'94)*, Santiago, Chile, Sep 1994.
- [Anderson and Horvitz, 2002] Corin R. Anderson and Eric Horvitz. Web montage: A dynamic personalized start page. In *Proceedings of the 11th World Wide Web Conference*, 2002.
- [Billsus and Pazzani, 1999] D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling (UM '99)*, Banff, Canada, 1999.
- [Chen *et al.*, 1996] M. Chen, J. Park, and P. Yu. Efficient data mining for path traversal patterns in distributed systems. In *Proc. of the 16th IEEE Intern'l Conf. on Distributed Computing Systems*, pages 385–392, May 1996.
- [Chen *et al.*, 1998] M. Chen, J. Park, and P. Yu. Efficient data mining for path traversal patterns. *IEEE Trans. on Knowledge and Data Engineering*, 10(2):209–221, Apr 1998.
- [Chi *et al.*, 2001] E. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions on the web. In *ACM CHI 2001 Conference on Human Factors in Computing Systems*, pages 490–497, Seattle WA, 2001.
- [Cooley *et al.*, 1999] Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava. Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1), 1999.
- [Duda and Hart, 1973] Richard Duda and Peter Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
- [Fayyad and Irani, 1993] U. Fayyad and K. Irani. Multi-interval discretization of continuous valued attributes for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI '93)*, pages 1022–1027, 1993.
- [Japkowicz, 2000] N. Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI '2000)*, 2000.
- [Jennings and Higuchi, 1993] Andrew Jennings and Hideyuki Higuchi. A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction*, 3(1):1–25, 1993.
- [John and Langley, 1995] George John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Eleventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Francisco, 1995. Morgan Kaufmann Publishers.
- [Ling and Li, 1998] C. Ling and C. Li. Data mining for direct marketing problems and solutions. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, New York, NY, 1998. AAAI Press.
- [Mobasher *et al.*, 1999] Bamshad Mobasher, R. Cooley, and J. Srivastava. Automatic personalization through web usage mining. Technical Report TR99-010, Department of Computer Science, DePaul University, 1999.
- [Perkowitz and Etzioni, 1997] Mike Perkowitz and Oren Etzioni. Adaptive sites: Automatically learning from user access patterns. Technical Report UW-CSE-97-03-01, University of Washington, 1997.
- [Porter, 1980] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, Jul 1980.
- [W3C,] W3C. Html 4.01 specification.
- [Zajicek and Powell, 1997] Mary Zajicek and Chris Powell. Building a conceptual model of the world wide web for visually impaired users. In *Proc. Ergonomics Society 1997 Annual Conference*, Grantham, 1997.
- [Zhu *et al.*, 2003] Tingshao Zhu, Russ Greiner, and Gerald Häubl. An effective complete-web recommender system. In *The Twelfth International World Wide Web Conference (WWW2003)*, Budapest, Hungary, May 2003.