An Effective Complete-Web Recommender System

Tingshao Zhu Russ Greiner
Department 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

There are a number of recommendation systems that can suggest the webpages, within a single website, that other (purportedly similar) users have visited. By contrast, our goal is a system that can recommend "information content" (IC) pages — i.e., pages that contain *information relevant to the user* — from *anywhere in the web*. This paper describes how we addressed this challenge, We first collected a number of annotated user sessions, whose pages each include a bit indicating whether it was IC. Our system, ICPF, then used this collection to learn the characteristics of words that appear in such IC-pages, in terms of the word's "browsing features" (e.g., did the user follow links whose anchor included this word, etc.).

This paper describes the ICPF system, as well as a tool (AIE) we developed to help users annotate their sessions, and a study we performed to collect these annotated sessions. We also present empirical data that validate the effectiveness of this approach.

1. INTRODUCTION

While the World Wide Web contains a vast quantity of information, it is often difficult for web users to find the information they really want. This paper presents a recommendation system, ICPF, that *identifies "information content" (IC) pages* — i.e., pages the user must examine to accomplish her[1] current task. Our system can locate these IC-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 [13, 14]. ICPF differs in several respects. First, as many recommendation systems are server-side, they can only provide information about one specific website, based on correlations amongst the pages that previous users have visited. By contrast, our client-side ICPF is not specific to a single website, but can point users to pages *anywhere in the Web*. The fact that our intended coverage is the entire Web leads to a second difference: *support*. As any single website has a relatively small number of pages, a website-specific recommendation system can expect many pages to have a large number of hits; it can therefore focus only on these high support (read "highly-visited") pages. Over the entire WWW, however, very few pages will have high support. Our system must therefore use a different approach to finding recommended pages, based on the users' abstract browsing patterns; see below. The third difference deals with the goal of the recommendation system: Many recommendation systems first determine other users that appear similar to the current user B, then recommend that B visit the pages that other similar users have visited. Unfortunately, there is no reason to believe that these correlated pages will contain information useful to B. Indeed, these suggested pages may correspond simply to irrelevant pages on the paths that others have taken towards their various goals, or worse, simply to standard dead-ends that everyone seems to hit. By contrast, our goal is to recommend only "information content" (IC) pages; i.e., pages that are essential to the user's task. To determine this, we first collected a set of annotated web logs (where the user has indicated which pages are IC), from which our learning algorithm learned to characterize the IC-page associated with the pages in any partial subsession. After examining the pages a user has visited in a (partial) session, our ICPF will then recommend only the associated IC-pages.

1.1 Overview of ICPF

Our goal is to help the user find "IC-pages" — i.e., pages that the user must examine to accomplish her task. This is clearly relative to her current information need; we estimate this from her current click-stream, based on properties of the *words* that appear there. That is, imagine the user has examined the URLs in the sequence $\langle U_1, U_2, \ldots, U_7 \rangle$, and observed that the word "workshop" appeared in each of the 4 most recent pages. Moreover, U_5 contained the results of a Google search (see Figure 6), which the user followed to U_6 (which is "Workshop Home Page"). Here, we observed that the title and snippet around U_5 also contained "workshop". We would associate the word "workshop" with the "browsing properties" that it appeared in (1) all 4 recent pages, (2) a result's title and (3) a snippet that was followed.

Our ICPF can use this browsing information to help determine IC-pages as it has earlier learned a model of user browsing patterns from previous annotated web logs (defined below).[2] In particular, imagine it had learned that

> If the word w appears in at least 2 recent pages,
> and w appears in a snippet that was followed, (1)
> then w will tend to appear in IC-pages.

Given this rule, ICPF would assert that "workshop" is likely to be an *IC-word* — i.e., a word that appears in an IC-page. ICPF would similarly compute browsing properties for essentially all of the words that appear in any of the pages $\{U_1, \ldots, U_7\}$, and then use this model to predict the user's current information need: a list

[1]We will use the female pronouns ("she" and "her") when referring to users, of either gender.

WWW2003, May 20–24, 2003, Budapest, Hungary.
ISBN 963-311-355-5.

[2]We use the term "ICPF" to refer to both the learner which produces a page classifier from training data (*viz.*, the annotated web logs), and also to the resulting classifier that predicts possible web pages from the user's current click stream; the classifier's interface is shown in Figure 1.

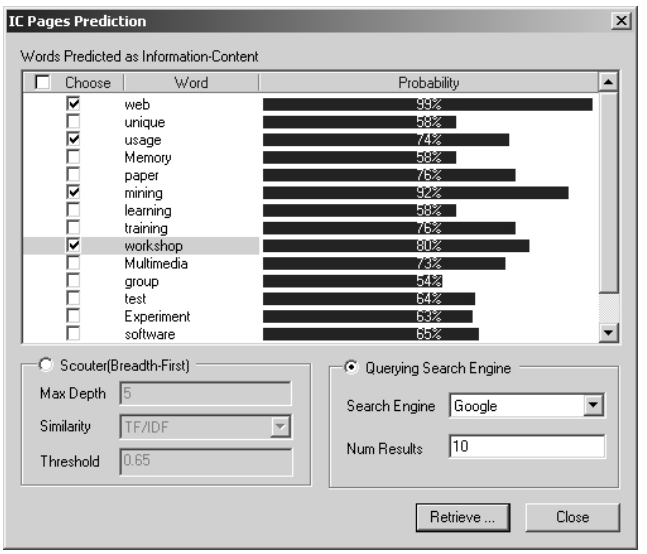

Figure 1: ICPF**: IC-page Prediction**

of word-probability pairs $\{ \langle w, p(w) \rangle \}$, where $p(w)$ estimates the probability that the word w will be an IC-word. It presents such information to the user; see the top portion of Figure 1. The check-boxes allow the user to identify which of the suggested words are actually part of her current information need. This figure shows two ways that ICPF could use this information: First, it could "scout ahead": follow the outward links from the current page (recursively, in a breadth-first fashion) seeking pages that include many of these IC-words. It would then recommend such IC-word-rich pages to the user. Alternatively, ICPF could send an appropriate query to a search engine (e.g., Google), then possibly scout forward from the pages returned. Our companion paper [20] discusses this issue in greater depth; it also considers whether these browsing-patterns are specific to individuals.

1.2 Outline

To build our ICPF system, we need client-side information, in the form of *annotated web-logs*: the sequence of webpages that users have visited, together with an "IC" label for each page: was this paper an IC-page or not? We use this data to train our ICPF, and later in our experiments, to verify that ICPF in fact worked effectively. *N.b.*, our performance system does *not* require the user to explicitly label pages as IC or not; this is done only from a control population, during the training phase.

Section 2 describes the client-side tool, AIE, that we developed for collecting this training data. AIE allows the user to explicitly indicate which pages were IC-pages for her specific current task. This section also describes the empirical study we ran to collect the data.

Section 3 shows how ICPF uses this collected information to learn an IC-word classifier: Given the user's current click stream, this classifier will predict which words will be in the IC-page. Here we discuss some challenges of dealing with this imbalanced dataset and show the results on this task. Section 4 evaluates this method, and shows that our ICPF works fairly well.

As mentioned above, there are many recommender systems that perform a related task. Section 5 discusses the most relevant of these systems, and describes how they differ from our objectives and approach.

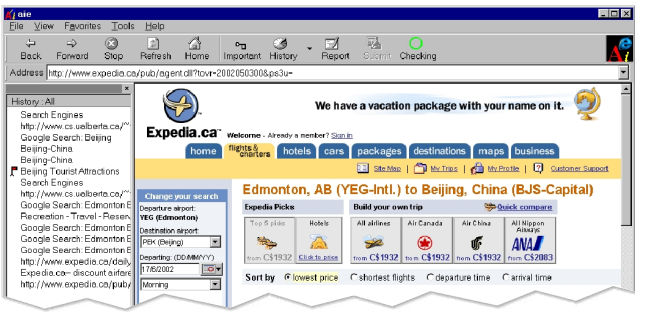

Figure 2: The AIE **Browser (top portion)**

2. AIE AND EMPIRICAL STUDY

2.1 Specific Task

To help us determine which pages are IC — i.e., contain information the user requires to complete her task — we first collected a set of *annotated web-logs*: each a sequence of webpages that a user has visited, and labeled with a bit that indicates whether she considered this page to be IC. We enlisted the service of a number of students (from the School of Business at the University of Alberta) to obtain these annotated web-logs. Each participant was asked to perform a specific task:

1. Identify 3 novel vacation destinations — i.e., places you have never visited.
2. Plan a *detailed* vacation to each destination specifying travel dates, flight numbers, accomodation (hotels, campsite, ...), activities, etc.

Each participant was given about 45 minutes, and given access to our augmented browsing tool (AIE; see Section 2.2), which recorded their specific web-logs, and required them to provide the "IC-page" annotation. The participants also had to produce a short report summarizing the vacation plans, which needed to explicitly cite the specific webpages ("IC-pages") that were involved in these decisions; here AIE made it easy to remember and insert these citations. To help motivate subjects to take this exercise seriously, we told them that two (randomly selected) participants would win $500 to help pay for the specific vacation they had planned.

We chose this specific task as

- It represents a fairly standard way of using the web
- It was goal-directed, in contrast to simply asking the participants to "meander about the web"
- The contents of many travel websites are fairly constant
- A diverse set of web-pages may be relevant — flight schedules, travel brochures, recent news (terrorist attacks), ...
- It is easy to motivate students to do this task, as they will get a chance to actually go on this trip
- The task is fairly well-defined and delimited

2.2 AIE: Annotation Internet Explorer

To enable us to collect the IC information, we built an enhanced version of Microsoft Internet Explorer, called AIE (shown in Figure 2), which we installed on all computers in the lab we used for our study. As with all browsers, the user can see the current web page. This tool incorporates several relevant extensions — see the toolbar across the top of Figure 2. First, the user can declare the

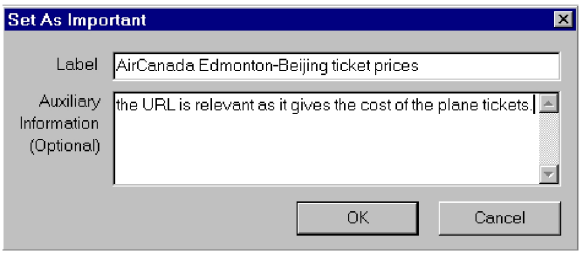

Figure 3: AIE **PopUp Window, used to declare a page as "IC"**

current page to be "important" (read "IC"), by clicking the *Important* button on the top bar. When doing this, AIE will pop up a new window (Figure 3) that shows this URL, and two fields that allow user input: a mandatory field requiring the user to enter an alias for this page (e.g., "AirCanada Edmonton-Beijing ticket prices") and an optional field for writing a short description of why this page was important (e.g., "the URL is relevant as it gives the cost of the plane tickets").

The *History* button on the toolbar brings up the side-panel (as shown in Figure 2), which shows the user the set of all pages seen so far, with a flag indicating which pages the user tagged as IC (important). The user can click on one of these to return to that IC-page; she can also reset its "importance" designation.

The *Report* button will switch fom the "Browse view" to the "Report editor", which participants can use to enter their report. Here, each subject has access to the pages she labeled as important during her browsing, which she can use in producing her report. (In fact, the participant can only use such pages in her report.[3])

After completing her report, the user can then submit her entire session using the *Submit* button. This sends over the entire sequence of websites visited, together with the user's "IC-page" annotations, as well as other information, such as time-stamps for each page, etc.[4]

2.3 Features of the Annotated Web Log Data

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 was 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 that each non-search-engine page was visited only 2.02 times on average. Figure 4 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. (See Section 5.)

[3]She does have the option of returning to the "browse" mode, and adding new pages to the list of important pages. She can also examine *all* pages visited earlier, and re-assign the pages — i.e., take a page considered non-IC, re-declare it to be IC, and then use that page in her report.

[4]In addition to collecting these sessions, we also downloaded a copy of every page visited by any of the participants, and we also explored the web site from the visited page down to 5 levels; this required about 9.1GB. We plan to use this later, to help analyse the structure of the websites visited, when they were visited.

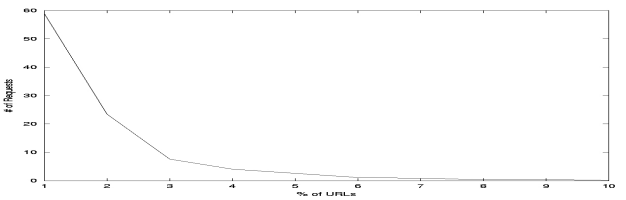

Figure 4: Frequency of URLs

3. LEARNING TASK

As motivated above, we are seeking general patterns that describe how the user locates useful information (IC-pages). For reasons described above (see also Section 5), 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. For example, if she follows a hyperlink to a page, but later backs up, this suggests that the snippet or anchor around the hyperlink contains words that seem very relevant (think "IC-words"), but the content of the page itself does not satisfy her information need.

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.

The previous section describes the source of our basic training data — annotated web logs. This section shows how we use that data to build a classifier for characterizing which *words* will appear in the IC-page. The first step (Section 3.1) 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 3.2 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 (see Section 3.3). By examining the associated IC-page, we also label each such word as an IC-word or not. Section 3.4 shows how our ICPF uses this information to train a classifier to predict when a word will be an IC-word.

3.1 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.

Chen et al. [3, 4] 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. [6] 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.

Algorithm ICSI:(URLsequence $U = \langle u_1, u_2, \ldots, u_n \rangle$):
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 5: ICSI **Algorithm**

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 5 summarizes our IC-session identification algorithm.

3.2 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.

3.3 Attribute Extraction

We consider all words that appear in all pages, removing stop words and stemming, using standard algorithms [15]. 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.)

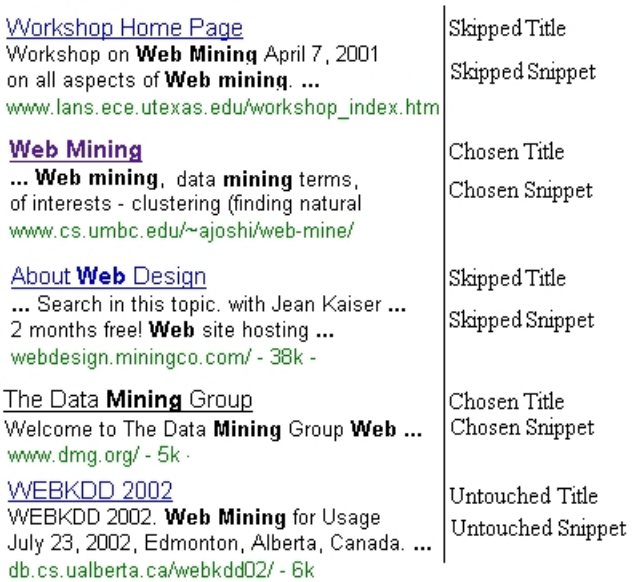

Figure 6: Title-Snippet State in the Search Result Page

3.3.1 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 6, 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*",[5] and all results after the last chosen link in the list will be "*Untouched*". Figure 6 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 *backward* if that page has been visited before; otherwise we say a page is *forward*.

The actual features used for each word w appear below. Each is with respect to a single IC-session. Notice that most have numeric scores and many are simple integers — e.g., how many times w is in some specified category.

[5]These are the links that the user probably saw, but actively chose not to follow.

isKeywordCnt Number of times that w appeared within the query's keyword list.

skippedTitleCnt Number of skipped titles containing w.

skippedSnippetCnt Number of skipped snippets that contain w.

chosenTitleCnt Number of chosen titles that include w.

chosenSnippetCnt Number of chosen snippets that include w.

untouchedTitleCnt Number of untouched titles that include w.

untouchedSnippetCnt Number of untouched snippets that include w.

unknownCnt Number of times that w appears in the anchor of a chosen link that is not one of the listed results — e.g., when the user clicks the hyperlink in the advertisement area.

bkTitleCnt Number of chosen titles that include w, but where the user later goes back to the same search result page, presumably to try another entry there. In Figure 6, the "Web Mining" entry is backed, as the user then went back to go to "The Data Mining Group".

bkSnippetCnt Number of chosen snippets that include w but were later "backed".

3.3.2 Sequential Attributes

All the following measures are extracted from the 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 occurences of w in the j^{th} "HTML context" [17]: and v_j is the weight associated with this context, shown as

h1	10	h6	5	strong	15	b	15	
h2	9	a	50	big	20	u	10	
h3	8	title	20	em	15	blink	20	(2)
h4	7	cite	10	i	15	s	5	
h5	6							

ratioWordAppearance Number of pages containing w divided by number of pages.

avWeight Average weight of w across the whole IC-session.

varWeight w's weight variation across the whole sequence.

trendWeight The trend of the word's weight in the whole sequence: { *ascend, descend, unchanged* }. If the word's weight becomes higher along the IC-session, it is expected to be IC-word with high probability.

ratioLinkFollow For the hyperlinks whose anchor text contain w, (followed hyperlinks whose anchor text contain w)/ (hyperlinks whose anchor text contain w).

ratioFollow How often w appeared in the anchor text of hyperlinks that followed — (number of followed hyperlinks whose anchor text contain w) / (length of IC-session - 1).

ratioLinkBack For the clicked hyperlinks whose anchor text contain w: (number of hyperlinks that were backed later) / (number of hyperlinks followed).

ratioBackward For these pages that contain w, (number of pages that are revisited) / (number of pages).

avWeightBackward The average weight of w in the backward pages.

varWeightBackward The variance of w's weight in the backward pages.

ratioForward For the pages that contain w, (number of pages that are forward) / (number of pages).

avWeightForward The average weight of w in the forward pages.

varWeightForward The variance of w's weight in the forward pages.

ratioInTitle For those pages that contain w , (number of pages that contain w in the title) / (number of such pages).

ratioInvisible For these pages that contain w, (number of pages where w is invisible) / (number of pages). We only count the words in META tags (keyword & description) as invisible.[6]

[6]We thought this might be very important, as many websites ensure that all the relevant words appear in the META elements of the page, as a way to help establish a good position in search engine results pages.

	SearchCnt	...	...	...	...	IC-Word
Web	1	1	...	0.7	...	Yes
Mining	1	1	...	0.9	...	Yes
resource	0	0	...	0.6	...	No
software	0	0	...	0.6	...	Yes
paper	0	0	...	0.2	...	No
...						

Figure 7: Feature Vector for some Extracted Words

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. Figure 7 shows the feature vectors for some words. 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 [19].) 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.

3.4 IC-word Prediction

After preparing the data, ICPF used Weka [18] to produce a NaïveBayes (NB) classifier. Recall that NB is a simple belief net structure which assumes that the attributes are independent of one another, conditioned on the class label [7]. NB also runs fast, and acquires the best performance compared to other classifiers, such as decision tree and Support Vector Machines (SVM). To deal with continuous attributes, we use estimation [8] instead of discretization [11], as we found the former works better.

4. EMPIRICAL RESULTS

Section 4.1 first provides a simple way to evaluate the quality of our predictions, determining whether we can predict the words on the IC-page based on the entire prior sequence. Section 4.2 then extends this to the general case; determining whether we can predict the IC-words well before then.

To train the classifer, we first identify IC-sessions (Figure 5), then extract the browsing properties of all the words in the IC-session except the last page, i.e., the IC-page. We use that final page to label each word as either IC or not. The classifier can then be trained to predict which words are IC-words given their browsing properties.

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

4.1 Simple Evaluation

According to the labelled log data, only 12.5% of the pages were IC. Moreover, the number of non-IC-words is far greater than that of IC-words. To deal with this imbalanced dataset, we have tried both down-sampling [12] and over-sampling [9], 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

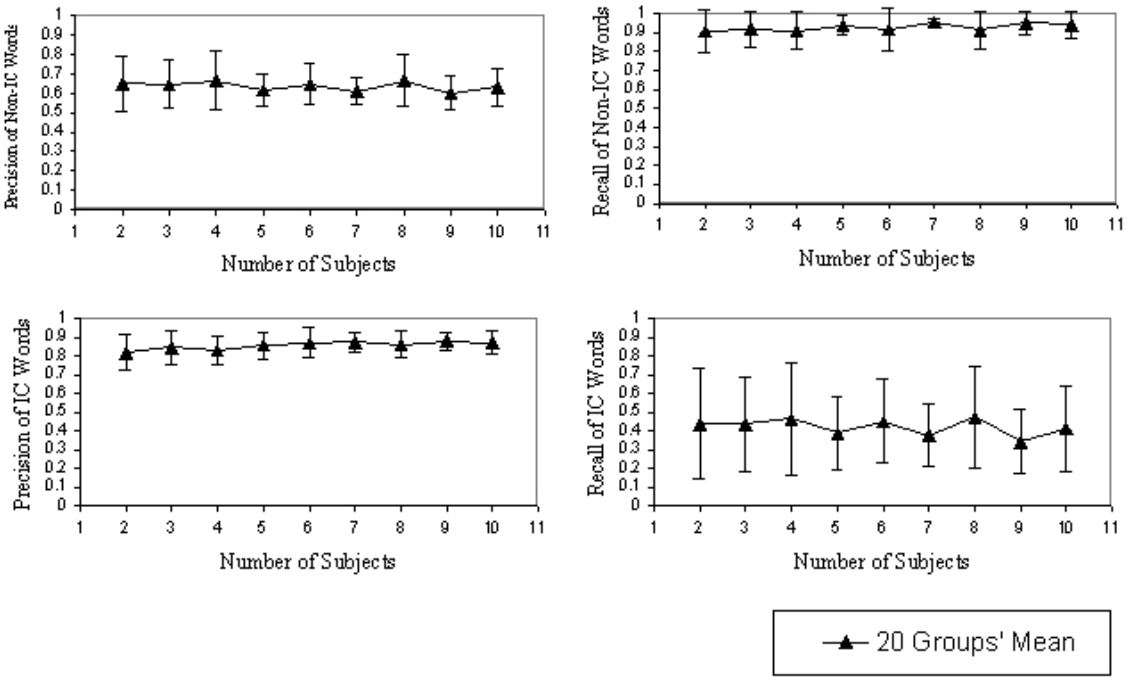

Figure 8: Non-IC and IC-word Prediction Results

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

For each $R = 2, 3, \ldots 10$, 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 that 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:

$$\begin{aligned} \text{ICprecision}(U) &= \frac{|WP(U_{N-1}) \bigcap W(u_N)|}{|WP(u_N)|} \\ \text{ICrecall}(U) &= \frac{|WP(U_{N-1}) \bigcap W(u_N)|}{|W(u_N)|} \\ \text{nonICprecision}(U) &= \frac{|\overline{WP(U_{N-1})} \bigcap \overline{W(U_{N-1})}|}{|\overline{WP(U_{N-1})}|} \\ \text{nonICrecall}(U) &= \frac{|\overline{WP(U_{N-1})} \bigcap \overline{W(U_{N-1})}|}{|\overline{W(U_{N-1})}|} \end{aligned} \quad (3)$$

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

We built 10-fold training/testing datasets for each of the 9×20 groups. For each of the 9 values of R, and each of the 4 quantities (Equation 3), we computed the median of the 20 associated values. (We used median as it is less sensitive to outliers than the mean.)

Figure 8 shows the means and standard deviations of these median values. The high precision and recall, in some cases, suggest that there is some commonality across users, which our algorithm is finding. Notice this is not based on one website or a specific set of words, but rather on how web users find useful information. This high level of generality means our model can be applied to different websites and different users. (Our companion paper [20] discusses the single-user case, which shows that information learned from a single user can be helpful to that user.)

Even though the average recall of IC-words is only about 45%, this is still good enough to find IC-pages, which of course is our ultimate goal. Section 1.1 presented two methods that ICPF can use to locate IC-pages given IC-words. 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, ICPF might try to build search queries using 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 that 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.)

4.2 General Evaluation Method

Clearly a good recommendation system should predict all-and-only the IC-pages. It would also be useful to predict these pages *early* — i.e., it is better to recommend the IC-page u_7 after the user has traversed only $\langle u_1, u_2 \rangle$, rather than wait until the user has visited all of $\langle u_1, u_2, u_3, \ldots, u_6 \rangle$, as this would save the user

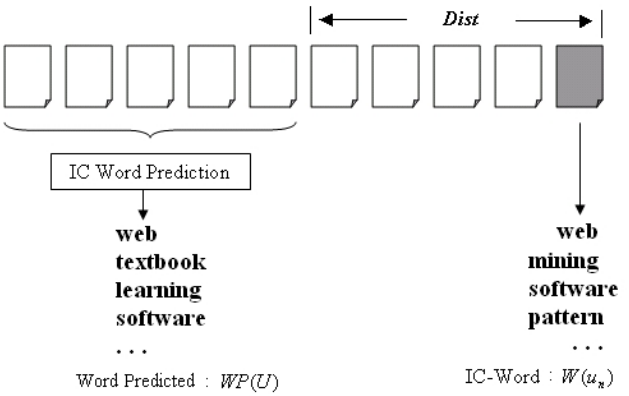

Figure 9: Evaluation Specification

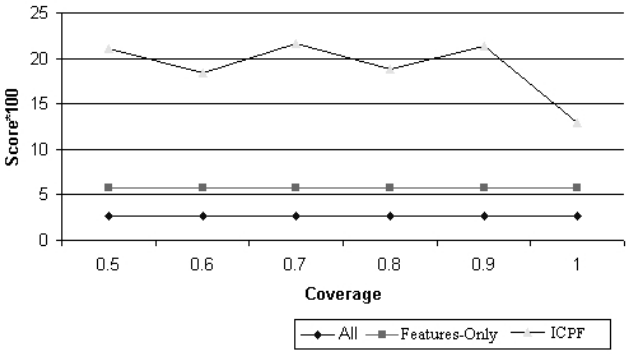

Figure 10: Evaluation Result

the need to visit the 4 intervening pages. We therefore define an evalution method based on these two objectives.

For each session $U = \langle u_1, u_2, u_3, \ldots, u_N \rangle$ of length N (where u_N is the IC-page), there are $N-1$ initial subsessions, where each subsession $U_\ell = \langle u_1, u_2, \ldots, u_\ell \rangle$ is the first consecutive ℓ pages, for $1 \leq \ell < N$.

We will call the recommendation function on each subsession U_ℓ to generate a propsed set of IC-words, $WP(U_\ell)$. We extend Equation 3 to be

$$\text{ICprecision}(U,\ell) = \frac{|WP(U_\ell) \bigcap W(u_N)|}{|WP(U_\ell)|}$$

$$\text{ICrecall}(U,\ell) = \frac{|WP(U_\ell) \bigcap W(u_N)|}{|W(u_N)|}$$

We then define the following F-Measure [16]

$$\text{F}(U,\ell) = \frac{2 \times \text{ICprecision}(U,\ell) \times \text{ICrecall}(U,\ell)}{\text{ICprecision}(U,\ell) + \text{ICrecall}(U,\ell)}$$

Using the fact that the distance between u_ℓ and u_N is $N-\ell$ (see Figure 9), we finally define

$$\textit{score}(U,\ell) = \begin{cases} |WP(U_\ell)| \times \textit{penalty} & \text{if } WP(U_\ell) \bigcap W(u_N) = \{\} \\ \frac{\text{F}(U,\ell)}{|WP(U_\ell)|} \times (N-\ell) & \text{otherwise.} \end{cases}$$

The *penalty* $\in \Re^-$ term basically penalizes the system for being silent; here we used -0.05. Notice this *score*$(\cdot,\cdot)$ increases the earlier the system can make a prediction, provided that prediction is accurate (based on the F-measure). We divide by the number of predicted words $|WP(U_\ell)|$ to discourage the system from simply suggesting everything.

For each IC-session U, let

$$\text{coverage}(U) = \frac{|W(u_N) \bigcap W(U_{N-1})|}{|W(u_N)|}$$

be the overlap between the words in the IC-page and the words in the other pages. (Notice *coverage*$(U) = 1$ corresponds to the $W(u_N) \subseteq W(U_{N-1})$ condition mentioned earlier.)

We randomly select 90% of the IC-sessions as training data, and use the others for testing. For each testing IC-session, we calculate the average *score* over all subsessions, provided there were any recommendations. (That is, we provide no recommendation if our system finds no word qualifies as an IC-word.) We then compute the average score for all testing IC-sessions as the final score for this trial. Figure 10 graphs this information, as a function of the coverage. We find that in most cases, while the coverage increases, $W(U_{N-1})$ grows very fast, which is the main reason that the score gets worse.

We compare our method with two other very simple techniques: let the IC-words be (1) all words in the subsession $WP(U_\ell)$, or (2) all feature words, which are those words enclosed by some specific HTML tags, such as "a", "title", "b", "h1", etc. Figure 10 shows that our approach did significantly better.

As a final observation about these data: When we compared the training data with the associated testing data, we found that only 30.77% of IC-pages in the testing data appeared anywhere in the training data; and that the average support for these in-training IC-pages is only 0.269. This is why we cannot use standard recommendation systems that only use page frequency: about 70% of the IC-pages would never be recommended, and even for the remaining 30%, there is only a small chance that they would be selected as recommendations; see below.

5. 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. [5] 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 is based on the idea 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 making an ad-hoc assumption.

One model, by Billsus and Pazzani [2], trained a NaïveBayes classifier 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

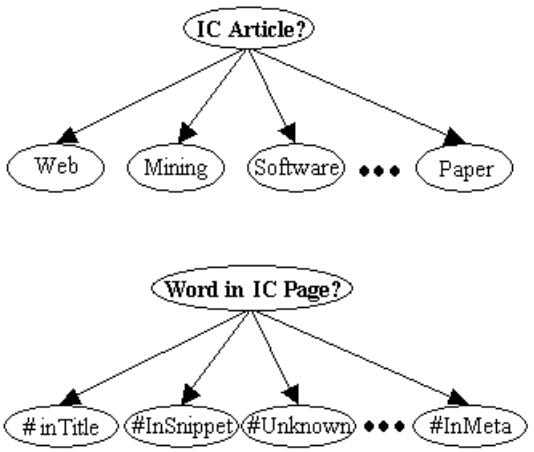

Figure 11: NaïveBayes Models for (a) IC-page Identification using Specific Words; (b) IC-word Identification using Word Features (our approach)

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 [10] trained one neural network for each user to represent a user's preferences 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.

In our research, 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 3.3. After training, our system may find some patterns like "any word that appears in the three consecutive pages will be in the IC-page"; see also Equation 1. Note this is different from systems that produce association rules [1] — 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 is also different 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 or 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. Figure 11 shows the difference between IC-page prediction based on specific words and our method based on word features.

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 [7] that classifies each webpage as IC or not, using a set of words as the features [2]. 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; see [20].

6. CONCLUSION

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 also plan to make a general version of AIE publicly available, to help us collect extensive data from different users, in natural environments.

We are implicitly viewing the list of IC-words as the intention of the web user. It would be tempting to ask users to indicate their own information need directly. However, they will often view this as an unwanted interruption and be reluctant to do so. In addition, it may not always be easy for individuals to express their information need. Our system is designed to run in the background, without any explicit user input. It observes a user's behavior and recommends IC-pages.

We are also 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 also 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.

Contributions

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 client-side recommendation system that can actually point a user to the webpages that contain the information that she will need to accomplish her current task, wherever those pages appear, 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. These become the features; our ICPF learns which combinations of these features determine whether the associated word is likely to be included in the IC-page. As our system only depends 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 labora-

tory 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.

Acknowledgments

The authors gratefully acknowledges the generous support from Canada's Natural Science and Engineering Research Council, the Alberta Ingenuity Centre for Machine Learning (http://www.aicml.org), and the Social Sciences and Humanities Research Council of Canada Initiative on the New Economy Research Alliances Program (SSHRC 538-2002-1013).

7. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] R. Cooley, B. Mobasher, and J. Srivastava. Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1), 1999.

[7] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[8] 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.

[9] N. Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI '2000)*, 2000.

[10] A. Jennings and H. Higuchi. A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction*, 3(1):1–25, 1993.

[11] G. John and P. 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.

[12] 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.

[13] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization through web usage mining. Technical Report TR99-010, Department of Computer Science, Depaul University, 1999.

[14] M. Perkowitz and O. Etzioni. Adaptive sites: Automatically learning from user access patterns. Technical Report UW-CSE-97-03-01, University of Washington, 1997.

[15] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, Jul 1980.

[16] C. Rijsbergen. *Information Retrieval*. 2nd edition, London, Butterworths, 1979.

[17] W3C. Html 4.01 specification.

[18] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, Oct. 1999.

[19] M. Zajicek and C. Powell. Building a conceptual model of the world wide web for visually impaired users. In *Proc. Ergonomics Society 1997 Annual Conference*, Grantham, 1997.

[20] T. Zhu, R. Greiner, and G. Häubl. Learning a model of a web user's interests. In *The 9th International Conference on User Modeling(UM2003)*, Johnstown, USA, June 2003.