Bringing Order to the Web: Automatically Categorizing Search Results

Hao Chen
School of Information Management & Systems
University of California
Berkeley, CA 94720 USA
hchen@sims.berkeley.edu

Susan Dumais
Microsoft Research
One Microsoft Way
Redmond, WA 99802 USA
sdumais@microsoft.com

ABSTRACT

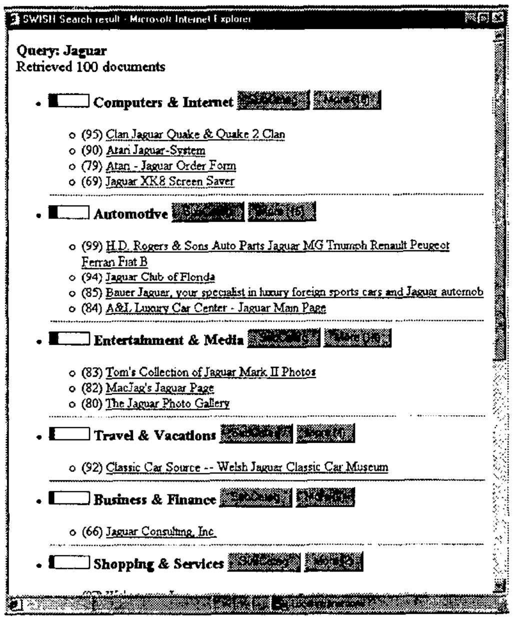
We developed a user interface that organizes Web search results into hierarchical categories. Text classification algorithms were used to automatically classify arbitrary search results into an existing category structure on-the-fly. A user study compared our new category interface with the typical ranked list interface of search results. The study showed that the category interface is superior both in objective and subjective measures. Subjects liked the category interface much better than the list interface, and they were 50% faster at finding information that was organized into categories. Organizing search results allows users to focus on items in categories of interest rather than having to browse through all the results sequentially.

Keywords
User Interface, World Wide Web, Search, User Study, Text Categorization, Classification, Support Vector Machine

INTRODUCTION

With the exponential growth of the Internet, it has become more and more difficult to find information. *Web search* services such as AltaVista, InfoSeek, and MSNWebSearch were introduced to help people find information on the web. Most of these systems return a ranked list of web pages in response to a user's search request. Web pages on different topics or different aspects of the same topic are mixed together in the returned list. The user has to sift through a long list to locate pages of interest. Since the 19^{th} century, librarians have used classification systems like Dewey and Library of Congress classification to organize vast amounts of information. More recently, *Web directories* such as Yahoo! and LookSmart have been used to classify Web pages. The manual nature of the directory compiling process makes it impossible to have as broad coverage as the search engines, or to apply the same structure to intranet or local files without additional manual effort.

To combine the advantage of structured topic information in directories and broad coverage in search engines, we built a system that takes the web pages returned by a search engine and classifies them into a known hierarchical structure such as LookSmart's Web directory [24]. The system consists of two main components: 1) a text classifier that categorizes web pages on-the-fly, and 2) a user interface that presents the web pages within the category structure and allows the user to manipulate the structured view (Figure 1).



Query: Jaguar
Retrieved 100 documents

- **Computers & Internet**
  - (95) Clan Jaguar Quake & Quake 2 Clan
  - (90) Atari Jaguar-System
  - (79) Atari - Jaguar Order Form
  - (69) Jaguar XK8 Screen Saver
- **Automotive**
  - (99) H.D. Rogers & Sons Auto Parts Jaguar MG Triumph Renault Peugeot Ferrari Fiat B
  - (94) Jaguar Club of Florida
  - (85) Bauer Jaguar, your specialist in luxury foreign sports cars and Jaguar automob
  - (84) A&L Luxury Car Center - Jaguar Main Page
- **Entertainment & Media**
  - (83) Tom's Collection of Jaguar Mark II Photos
  - (82) MacJag's Jaguar Page
  - (80) The Jaguar Photo Gallery
- **Travel & Vacations**
  - (92) Classic Car Source -- Welsh Jaguar Classic Car Museum
- **Business & Finance**
  - (66) Jaguar Consulting, Inc.
- **Shopping & Services**

Figure 1: Presenting web pages within category structure

RELATED WORK

Generating structure

Three general techniques have been used to organize documents into topical contexts. The first one uses *structural information* (meta data) associated with each document. The DynaCat system by Pratt [15] used meta data from the UMLS medical thesaurus to organize search results. Two prototypes developed by Allen [1] used meta data from the Dewey Decimal System for organizing results. In the SuperBook project [10], paragraphs of texts were organized into an author-created hierarchical table of contents. Marchionini et al. [12] also used table of content

CHI '2000 The Hague, Amsterdam
Copyright ACM 2000 1-58113-216-6/00/04...$5.00

views for structuring information from searches in the Library of Congress digital library. Others have used the link structure of Web pages to automatically generate structured views of Web sites. Maarek et al.'s WebCutter system [11] displayed a site map tailored to the user's search query. Wittenburg and Sigman's AMIT system [18] showed search results in the context of an automatically derived Web site structure. Chen et al.'s Cha-Cha system [4] also organized search results into automatically derived site structures using the shortest path from the root to the retrieved page. Manually-created systems are quite useful but require a lot of initial effort to create and are difficult to maintain. Automatically-derived structures often result in heterogeneous criteria for category membership and can be difficult to understand.

A second way to organize documents is by *clustering*. Documents are organized into groups based their overall similarity to one another. Zamir et al. [19, 20] grouped Web search results using suffix tree clustering. Hearst et al. [7, 8] used the scatter/gather technique to organize and browse documents. One problem with organizing search results in this way is the time required for on-line clustering algorithms. Single-link and group-average methods typically take $O(n^2)$ time, while complete-link methods typically take $O(n^3)$, where n is the number of documents returned. Linear-time algorithms like k-means are more efficient being $O(nkT)$, where k is the number of clusters and T the number of iterations. In addition, it is difficult to describe the resulting clusters to users. Clusters are usually labeled by common phrases extracted from member documents, but it is often difficult to quickly understand the contents of a cluster from its label.

A third way to organize documents is by *classification*. In this approach, statistical techniques are used to learn a model based on a labeled set of training documents (documents with category labels). The model is then applied to new documents (documents without category labels) to determine their categories. Chakrabarti et al. [2], Chekuri [3], and Mladenic [13] have developed automatic classifiers for subsets of pages from the Yahoo! Web directory. Only a small number of high level categories were used in their published results. And, the focus of these papers was on the underlying text classification algorithms and not on user interfaces that exploit the results. Recently, Inktomi [22] announced that it had developed techniques for automatic classification of web pages. However, its technical details were not disclosed and we are not aware of any search services employing this technology.

Using structure to support search

A number of web search services use category information to organize the search results. Yahoo! [27], Snap [26] and LookSmart [24] show the category label associated with each retrieved page. Results are still shown as a ranked list with grouping occurring only at the lowest level of the hierarchy (for Yahoo! and Snap). There is, for example, no way to know that 70% of the matches fell into a single top-level category. In addition, these systems require pre-tagged content. Before any new content can be used, it must be categorized by hand. Northern Light [25] provides Custom Folders in which the retrieved documents are organized hierarchically. The folders are organized according to several dimensions -- source (sites, domains), type (personal page, product review), language, and subject. Individual categories can be explored one at a time. But, again no global information is provided about the distribution of search results across categories.

The most common interface for manipulating hierarchical category structures is a hierarchical tree control, but other techniques have been explored as well. Johnson et al. [9] used a treemap that partitioned the display into rectangular bounding boxes representing the tree structure. Characteristics of the categories and their relationships were indicated by their sizes, shapes, colors, and relative positions. Shneiderman et al. [17] have recently developed a two-dimensional category display that uses categorical and hierarchical axes, called hieraxes, for showing large results sets in the context of categories. Hearst et al. [5] used three-dimensional graphics to display categories together with their documents. Multiple categories could be displayed simultaneously along with their hierarchical context. In all of these systems, documents must have pre-assigned category tags.

Few studies have evaluated the effectiveness of different interfaces for structuring information. Landauer et al. [10] compared two search interfaces for accessing chemistry information -- SuperBook which used a hierarchical table of contents, and PixLook which used a traditional ranked list. Browsing accuracy was higher for SuperBook than PixLook. Search accuracy and search times were the same for the two interfaces. However, different text pre-processing and search algorithms were used in the two systems so it is difficult to compare precisely. More recently, Pratt et al. [16] compared DynaCat, a tool that automatically categorized results using knowledge of query types and a model of domain terminology, with a ranked list and clustering. Subjects liked DynaCat's category organization of search results. Subjects found somewhat more new answers using DynaCat, but the results were not reliable statistically, presumably because there were only 15 subjects and 3 queries in the experiment.

In this paper we describe a new system showing how automatic text classification techniques can be used to organize search results. A statistical text classification model is trained offline on a representative sample of Web pages with known category labels. At query time, new search results are quickly classified on-the-fly into the learned category structure. This approach has the benefit of using known and consistent category labels, while easily incorporating new items into the structure. The user interface compactly displays web pages in a hierarchical category structure. Heuristics are used to order categories and select results within categories for display. Users can further expand categories on demand. Tooltip-like overlays are used to convey additional information about individual web pages or categories on demand. We compared our category interface with a traditional list interface under exactly the same search conditions. We now describe each of these components in more detail.

TEXT CLASSIFICATION

Text classification involves a training phase and a testing phase. During the training phase, web pages with known category labels are used to train a classifier. During the testing or operational phase, the learned classifier is used to categorize or tag new web pages.

Data Set

For training purposes, we used a collection of web pages from LookSmart's Web directory [24]. LookSmart's directory is created and maintainted by 180 professional Web editors. For our experiments, we used the directory as it existed in May 1999. At that time there were 13 top-level categories, 150 second-level categories, and over 17,000 categories in total. On average each web page was classified into 1.2 categories.

Pre-processing

A text pre-processing module extracted plain text from each web page. In addition, the title, description, keyword, and image tag fields were also extracted if they existed. A vector was created for each page indicating which terms appeared in that page.

The results returned by search engines contain a short summary of information about each result. Although it is possible to download the entire contents of each web page, it is too time consuming to be applicable in a networked environment. Therefore, in our prototype, the initial training and subsequent classification are performed using only summaries of each web page. The training summaries were created using the title, the keyword tag, and either the description tag if it existed or the first 40 words otherwise. When classifying search results we use the summary provided in the search results.

Classification

A Support Vector Machine (SVM) algorithm was used as the classifier, because it has been shown in previous work to be both very fast and effective for text classification problems [5][14]. Roughly speaking, a linear SVM is a hyperplane that separates a set of positive examples (i.e., pages in a category) from a set of negative examples (i.e., pages not in the category). The SVM algorithm maximizes the margin between the two classes; other popular learning algorithms minimize different objective functions like the sum of squared errors. Web pages were pre-processed as described above. For each category we used the 1000 terms that were most predictive of the category as features. Vectors for positive and negative examples were input into the SVM learning algorithm. The resulting SVM model for each category is a vector of 1000 terms and associated weights that define the hyperplane for that category.

We used 13,352 pre-classified web pages to train the model for the 13 top-level categories, and between 1,985 and 10,431 examples for each of these categories to train the appropriate second-level category models. The total time to learn all 13 top-level categories and 150 second-level categories was only a few hours. Once the categories are learned, the results from any user query can be classified. At query time, each page summary returned by the search engine is compared to the 13 top-level category models. A page is placed into one or more categories, if it exceeds a pre-determined threshold for category membership. Pages are classified into second-level categories only on demand using the same procedure.

We explored a number of parameter settings and text representations and used the optimal ones for classification in our experiment. Our fully automatic methods for assigning category labels agreed with the human-assigned labels almost 70% of the time. Most of the disagreements were because additional labels were assigned (in addition to the correct one), or no labels were assigned. This is good accuracy given that we were working with only short summaries and very heterogeneous web content. Although classification accuracy is not perfect, we believe it can still be useful for organizing Web search results.

USER INTERFACE

The search interface accepted query keywords, passed them to a search engine selected by the user, and parsed the returned pages. Each page was classified into one or more categories using the learned SVM classifier. The search results were organized into hierarchical categories as shown in Figure 1. Under each category, web pages belonging to that category were listed. The category could be expanded (or collapsed) on demand by the user. To save screen space, only the title of each page was shown (the summary can be viewed by hover text, to be discussed later). Clicking on the title hyperlink brought up the full content of the web page in another browser window, so that the category structure and the full-text of pages were simultaneously visible.

Information Overlays

There is a constant conflict between the large amount of information we want to present and the limited screen real estate. We presented the most important information (titles of web pages and category labels) as text in the interface, and showed other information using small icons or transient visual overlays. The techniques we used included:

- A partially filled green bar in front of each category label showed the percentage of documents falling into the category. This provided users with an overview of the distribution of matches across categories.

- We presented additional category information (parent and child category labels) as hover text when the mouse hovered over a category title. This allowed users to see the subcategories for category as well as the higher-level context for each page.
- The summaries of the web pages returned by search engines provide users with additional information about the page helping them decide which pages to explore in greater depth. In order to present category context along with the search results, we displayed only titles by default and showed summaries as hover text when the mouse hovered over the titles of web pages.

Distilled Information Display

Even with the help of information overlays, there is still more information than a single screen can accommodate. We developed heuristics to selectively present a small portion of the most useful information on the first screen. The first screen is so important that it usually determines whether the user will continue working on this search or abandon it all together. We wanted to enable the user to either find the information there or identify a path for further exploration. In order to do this effectively we must decide: how many categories to present, how many pages to present in each category, how to rank pages within a category, and how to rank categories.

We presented only top-level categories on the first screen. There were several reasons for this. First, the small number of top level categories helped the user identify domains of interest quickly. Second, it saved a lot of screen space. Third, classification accuracy was usually higher in top level categories. Fourth, it was computationally faster to match only the top-level categories. Fifth, subcategories did not help much when there were only a few pages in the category. The user can expand any category into subcategories by clicking a button.

In each category, we showed only a subset of pages in that category. We decided to show a fixed number of pages (20) across all categories, and divided them in proportion to the number of pages in that category. So, if one category contained 50% of results, we would show 10 pages from that category in the initial view. The user can see all pages in a category by clicking a button.

Three parameters affected how pages are ordered within a category: its original ranking order in the results, its match score (if returned by the search engine), and the probability that it belongs to the category according to the classifier. For the experiment, we used only the rank order in the original search results to determine the order of items within each category. Thus if all the search result fall into one category the category organization returns the same items in the same order as the ranked list.

The categories can be ordered either in a static alphabetical order, or dynamically according to some importance score. The advantage of dynamic ranking is to present the most likely category first. The disadvantage is that it prevents the user from establishing a mental model of the relative position of each category in the browser window. For our experiment, importance was determined by the number of pages in the category. The category with the most items in it was shown first, and so on.

USER STUDY

A user study was conducted to compare the category-based interface (referred to as "Category Interface" henceforth) with the conventional search interface where pages are arranged in a ranked list (referred to as "List Interface" henceforth). The two interfaces are shown in Figure 2.

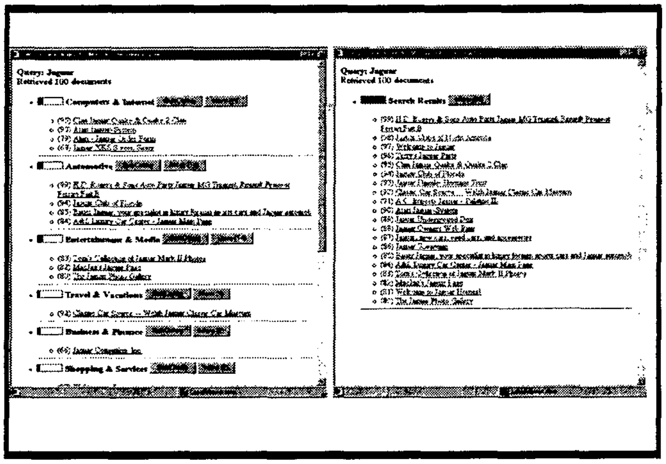


Figure 2: Category vs. List Interface

The top 100 search results for the query *"jaguar"* are used in this example. Twenty items are shown initially in both interfaces. In the List interface the 20 items can be seen without scrolling; in the Category interface scrolling is always required in spite of our attempt to conserve screen space. In both interfaces, summaries are shown on hover. Both interfaces contain a ShowMore button which is used to show the remaining items in the category; in the case of the List interface the remaining 80 items are shown. In addition, in the Category interface a SubCategory button is used to sub-categorize the pages within that category. The same control program is used in both cases, so timing is the same in both interfaces.

Methods

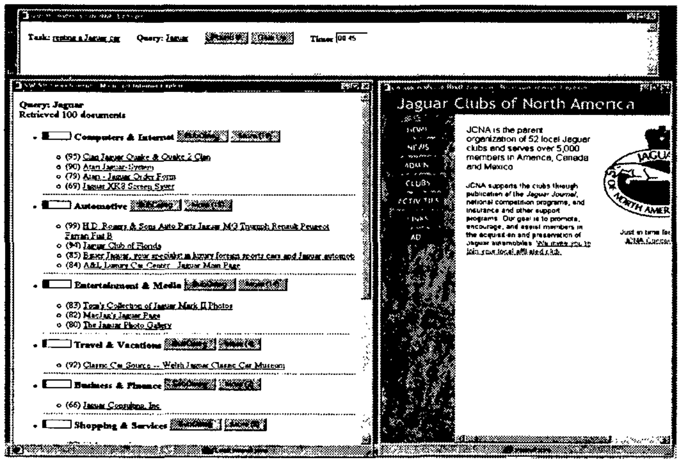
Subjects

Eighteen subjects of intermediate web ability participated in the experiment. Subjects were adult residents of the Seattle area recruited by the Microsoft usability lab, and represent a range of ages, backgrounds, jobs and education level.

Procedure

The experiment was divided into two sessions with a voluntary break between. Subjects used the Category interface in one session and the List interface in the other. The user read a short tutorial before each session began. During each session, the user performed 15 web search tasks, for a total of 30 search tasks. At the end of the experiment, the user completed an online questionnaire giving his/her subjective rating of the two interfaces. The total time for the experiment was about 2 hours.

During the experiment, the subject worked with three windows (Figure 3). The control window on the top shows the task and the query keywords. In this example, the ***task*** is to find out about *"renting a Jaguar car"* and the ***query*** we automatically issued is *"jaguar"*. The search results were displayed in the left bottom window. In the Category interface, the results were automatically organized into different categories, and in the List interface, the top 20 items were shown on the initial screen.



Task: renting a Jaguar car Query: Jaguar Timer: 00 45

Query: Jaguar
Retrieved 100 documents

- Computers & Internet
  - (95) Clan Jaguar Quake & Quake 2 Clan
  - (90) Atari Jaguar-System
  - (79) Atari - Jaguar Order Form
  - (69) Jaguar XK8 Screen Saver
- Automotive
  - (99) H.D. Rogers & Sons Auto Parts Jaguar MG Triumph Renault Peugeot Ferrari Fiat B
  - (94) Jaguar Club of Florida
  - (85) Baker Jaguar, your specialist in luxury foreign sports cars and Jaguar automob
  - (84) A&L Luxury Car Center Jaguar Main Page
- Entertainment & Media
  - (83) Tom's Collection of Jaguar Mark II Photos
  - (82) MacJag's Jaguar Page
  - (80) The Jaguar Photo Gallery
- Travel & Vacations
  - (92) Classic Car Source -- Welsh Jaguar Classic Car Museum
- Business & Finance
  - (66) Jaguar Consulting, Inc.
- Shopping & Services

Jaguar Clubs of North America

JCNA is the parent organization of 52 local Jaguar clubs and serves over 5,000 members in America, Canada and Mexico

JCNA supports the clubs through publication of the *Jaguar Journal*, national competition programs, and insurance and other support programs. Our goal is to promote, encourage, and assist members in the acquisition and preservation of Jaguar automobiles. We invite you to join your local affiliated club.

Figure 3: Screen of the User Study

When the subject clicked on a hyperlink, the page opened in the right window. When the subject found an answer, s/he clicked on the "Found It!" button in the control window. If no answer could be found, s/he clicked on the "Give Up" button. There was a timer in the control window that reminded the subject after five minutes had passed. If a reminder occurred, the subject could continue searching or move on to the next task. User events such as hovering over a hyperlink to read the summary, clicking on a hyperlink to read the page, expanding or collapsing the list were logged.

Search Tasks

The 30 search tasks were selected from a broad range of topics, including sports, movies, travel, news, computers, literature, automotive, local interest, etc. Ten of the queries were popular queries from users of MSNWebSearch. In order to facilitate evaluation we selected tasks that had reasonably unambiguous answers in the top 100 returned pages (a kind of known-item search). The tasks varied in difficulty – 17 had answers in the top 20 items returned (on the first page in the List interface), and 13 had answers between ranks 21 and 100. The tasks also varied in how much manipulation was required in the Category interface – 10 required subjects to use ShowMore or SubCategory expansion, and 10 required some scrolling because the correct category was not near the top.

To ensure that results from different subjects were comparable, we fixed the keywords for each query in the experiment. We also cached the search results before the experiments so that each subject got the same results for the same query. The MSNWebSearch engine [22] was used to generate the search results.

Each subject performed the same 30 search tasks. For 15 tasks they used the Category interface and for 15 they used the List interface. The order in which queries were presented and whether the Category or List interface was used first was counterbalanced across subjects. Nine lists of tasks were used -- each list contained all the tasks in a different order and was assigned to a pair of subjects, one in the Category-first condition and one in the List-first condition. This yoking of presentation orders reduces error variance which is desirable given the relatively small number of subjects and tasks we used.

Results

The main independent variable is the Category interface vs. the List interface. The order of presentation (List first or Category first) is a between subject variable. We analyzed both subjective questionnaire measures and objective measures (search time, accuracy, and interactions with the interface such as hovering, and displaying Web pages).

Subjective questionnaire measures

After the experiment, subjects completed a brief online questionnaire. The questionnaire covered prior experience with Web searching, ratings of the two interfaces (on a 7-point scale), and open-ended questions about the best and worst aspects of each interface. Seventeen of the eighteen subjects used the Web at least every week, and eleven of the eighteen subjects searched for information on the Web at least every week. The most popular Web search service among our subjects was Yahoo!.

Subjects reported that the Category interface was "easy to use" (6.4 vs. 3.9, $t(17) = 6.41$; $p \ll 0.001$), they "liked using it" (6.7 vs. 4.3, $t(17) = 6.01$; $p \ll 0.001$), they were "confident that I could find the information if it was there" (6.3 vs. 4.4, $t(17) = 4.91$; $p \ll 0.001$), that it was "easy to get a good sense of the range of alternatives" (6.4 vs. 4.2, $t(17) = 6.22$; $p \ll 0.001$), and that they "prefer this to my usual search engine" (6.4 vs. 4.3, $t(17) = 4.13$; $p \ll 0.001$). On all of our overall measures subjects much preferred the Category interface.

For the two questions that asked about the usefulness of interface features (hover text and ShowMore), there were no reliable differences between interfaces, suggesting that subjects did not simply have an overall positive bias in responding to questions about the Category interface. Subjects thought the display of page summaries in hover text was useful in both interfaces (6.5 Category vs. 6.4 List, t(17) = 0.36; p<0.72), and that the ShowMore option was useful (6.5 Category vs. 6.1 List, t(17) = 1.94; p<0.07).

Accuracy/GiveUp.

When creating search tasks, we had a target correct answer in mind. However, other pages might be relevant as well, so we examined all pages that subjects said were relevant to see if they in fact answered the search task. We looked at performance with strict and liberal scoring of accuracy. For strict scoring only pages that were deemed by the experimenters to be relevant (after including additional pages found by subjects that we had missed) were counted as relevant. Using the strict criterion, there were slightly more wrong answers in the List interface (1.72 out of 30) than in the Category interface (1.06 out of 30), but this difference is not reliable statistically using a paired t-test (t(17) = -1.59; p<.13). The lack of difference between interfaces is not surprising, since it reflects a difference in criterion about what the correct answer is rather than task difficulty per se. For liberal scoring, any answer that subjects said was relevant was deemed relevant, so by definition, there were no wrong answers in either interface. We used the liberal scoring in subsequent analyses.

Subjects were allowed to give up if they could not find an answer. They could do this at any time during a trial. After 5 minutes had elapsed for a task, subjects were notified and encouraged to move onto the next task. Some subjects continued searching, but most gave up at this time. There are significantly more tasks on which subjects gave up in the List interface than in the Category interface (t(17) = -2.41; p<.027), although the absolute number of failures is small in both interfaces (0.77 in List and 0.33 in Category).

Search Time

We used the median search time across queries for statistical tests, because reaction time distributions are often skewed and statistical tests can be influenced by outliers. (We also find exactly the same results using mean reaction times, so outliers were not a problem in this experiment.) A 2x2 mixed design was used to measure differences in search time. The between subjects factor is whether subjects saw the List or Category interface first, and the within subjects factor is List or Category interface. Median search times are shown in Figure 4.

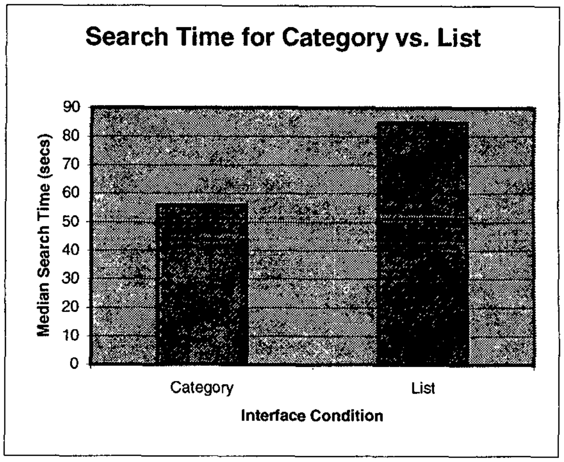


Figure 4: Search time by interface type

There is a reliable main effect of interface type, with a median response time of 56 seconds for the Category interface and 85 seconds for the List interface (F(1,16) = 12.94; p=.002). The advantage is not due to a speed-accuracy tradeoff or to a tendency to give up on difficult queries, since if anything subjects in the Category interface were more accurate (when scored strictly) and gave up less often. This is a large effect both statistically and practically. It takes subjects 50% longer to find answers using the List interface. On average it took subjects 14 minutes to complete 15 tasks with the Category interface, and 21 minutes with the List interface. There is no effect of the order in which interfaces were shown, list first or category first (F(1,16) = 0.26; p=0.62). And, there is no interaction between order and interface (F(1,16) = 1.23; p=0.28), which shows that results are not biased by order of presentation.

There are large individual differences in search time. The fastest subject finished the 30 search tasks in a median of 37 seconds, and the slowest in 142 seconds. But, the advantage of the Category interface is consistent across subjects.

There are also large differences across tasks or queries. The easiest task was completed in a median of 22.5 seconds, and the most difficult task required 166 seconds to complete. We divided the queries into those whose answers were on the first screen of the List interface (i.e., in the Top20 returned by the search engine) and those whose answers were not in the Top20. The search times are shown in Figure 5. Not surprisingly, there is a reliable main effect of whether the answer is in the Top20 or not -- median time for Top20 (57 seconds) and NotTop20 (98 seconds), F(1,56) = 16.5; p<<.001.

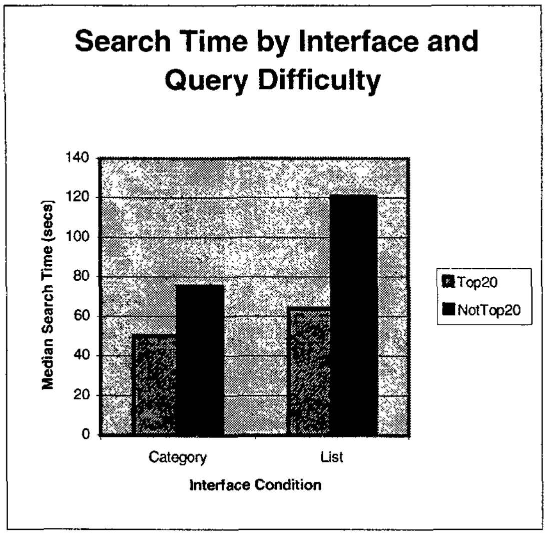


Figure 5: Search time by interface type and query difficulty

There is no interaction between query difficulty and interface (F(1,56)=2.52; p=.12). The Category interface is beneficial for both easy and hard queries. Although there is a hint that the category interface is more helpful for difficult queries, the interaction is not reliable. The Category interface is still beneficial even when the matching web page is in the first page of results. In our List interface items which were in the Top20 did not require any scrolling, whereas several of the Category interfaces for these items did. The advantage appears to be due to the way in which the category interface breaks the list of returned items down into easily scanable semantic chunks.

Interaction Style – Hovering, Page Views, ShowMore, SubCategory

We measured the number of hovering and page viewing actions subjects performed in the course of finding the answers. Subjects in the List interface hovered on more items than those in the Category interface (4.60 vs. 2.99; t(17) = -5.54; p<<0.001). The number of pages that subjects actually viewed in the right window is somewhat larger in the List interface (1.41 List vs. 1.23 Category; t(17) = -2.08; p<0.053). Although the difference is not large, it suggests that the category structure can help disambiguate the summary in the hover text. It is interesting to note that the average number of page views is close to 1, suggesting that users could narrow down their search by reading just the titles and summaries. Subjects read the full pages mostly to confirm what they found in the summary. This significantly reduces search time because the short summaries can be read faster than a full page of text, and there is no network latency for accessing summaries (summaries were stored locally, but retrieving the full-text of the pages required net access).

We also measured the expansion operations that subjects used in searching for information. In the List interface, subjects could expand this list of results by ShowMore. In the Category interface, subjects could ShowMore within each category, or they could break down categories into SubCategories. Overall, subjects in the Category interface used more expansion operations (0.78 ShowMore + SubCategories in Category vs. 0.48 Show More in List; t(17) = 3.54; p<0.003). So, subjects performed more expansion operations in the Category interface, but the selective nature of the operations (i.e., they applied to only a single category) meant that they were nontheless more efficient overall in finding things.

CONCLUSION

We developed and evaluated a user interface that organizes search results into a hierarchical category structure. Support Vector Machine classifiers were built offline using manually classified web pages. This model was then used to classify new web pages returned from search engines on-the-fly. This approach has the advantage of leveraging known and consistent category information to assist the user in quickly focusing in on task-relevant information. The interface allows users to browse and manipulate categories, and to view documents in the context of the category structure. Only a small portion of the most important and representative information is displayed in the initial screen, and hover text and overlay techniques are used to convey more detailed information on demand. A user study compared the category interface with traditional list interface using the same set of tasks, search engine, and search results. The results convincingly demonstrate that the category interface is superior to the list interface in both subjective and objective measures.

There are many directions for further research. One issue to explore is how the results generalize to other domains and task scenarios. The categories used in our experiment were designed to cover the full range of Web content. Nonetheless, not all user queries will match the category structure to the same extent. Results for some queries may fall entirely within one category (e.g., results for the query *"used parts for Jaguar XJ6L"*, would likely fall entirely within the Automobile category). In such cases, the Category interface (given our current display heuristics) is exactly the same as the List interface, so we are no worse off. Results for other queries may not match any of the categories very well. In our current interface we have a "NotCategorized" group at the bottom. In our experiment 5-40% of the results for each query were NotCategorized, but few of the answers were in the NotCategorized group. We hope to deploy our system more widely to look at this issue by getting a large sample of typical user queries. This would also allow us to explore a wider range of user tasks in addition to the known-item scenario we used.

There are also many interesting issues concerning how best to present concise views of search results in their category contexts. We chose to order categories by the number of matches and within each category to order the pages by search rank. Our text classification algorithms can easily handle thousands of categories, and we may have to move beyond our simple display heuristics for such cases.

ACKNOWLEDGMENTS

We are grateful to John Platt for help with the Support Vector Machine code, to Kirsten Risden for help in setting up the user study, and to reviewers for helpful suggestions.

REFERENCES

1. Allen, R. B., Two digital library interfaces that exploit hierarchical structure. In *Proceedings of DAGS95: Electronic Publishing and the Information Superhighway* (1995).
2. Chakrabarti, S., Dom, B., Agrawal, R., and Raghavan, P. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *The VLDB Journal 7*, (1998), 163-178.
3. Chekuri, C., Goldwasser, M., Raghavan, P. and Upfal, E. Web search using automated classification. In *Sixth International World Wide Web Conference*, Santa Clara, California, Apr. 1997, Poster POS725.
4. Chen, M., Hearst, M., Hong, J., and Lin, J. Cha-Cha: a system for organizing intranet search results. In *Proceedings of the 2nd USENIX Symposium on Internet Technologies and SYSTEMS (USITS)* (Boulder CO, October 1999) (to appear).
5. Dumais, S. T., Platt, J., Heckerman, D. and Sahami, M. Inductive learning algorithms and representations for text categorization. In *Proceedings of ACM-CIKM98*, Nov. 1998.
6. Hearst, M., and Karadi, C. Searching and browsing text collections with large category hierarchies. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI), Conference Companion* (Atlanta GA, March 1997).
7. Hearst, M., and Pedersen, P. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of 19th Annual International ACM/SIGIR Conference* (Zurich 1996).
8. Hearst, M., Pedersen, J., and Karger, D. Scatter/gather as a tool for the analysis of retrieval results. *Working Notes of the AAAI Fall Symposium on AI Applications in Knowledge Navigation* (Cambridge MA, November 1995).
9. Johnson, B., and Shneiderman, B. Treemaps: a space-filling approach to the visualization of hierarchical information structures. In *Sparks of Innovation in Human-Computer Interaction.* Ablex Publishing Corporation, Norwood NJ, 1993
10. Landauer, T., Egan, D., Remde, J., Lesk, M., Lochbaum, C., and Ketchum, D. Enhancing the usability of text through computer delivery and formative evaluation: the SuperBook project. In *Hypertext – A Psychological Perspective.* Ellis Horwood, 1993.
11. Maarek, Y., Jacovi, M., Shtalhaim, M., Ur, S., Zernik, D., and Ben Shaul, I.Z. WebCutter: a system for dynamic and tailorable site mapping. In *Proceedings of the 6th International World Wide Web Conference* (Santa-Clara CA, April 1997).
12. Marchionini, G., Plaisant, C., and Komlodi, A. Interfaces and tools for the Library of Congress national digital library program. *Information Processing and Management, 34*, 535-555, 1998.
13. Mladenic, D. Turning Yahoo into an automatic web page classifier. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI'98)* 473-474.
14. Platt, J. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods –Support Vector Learning*. B. Schölkopf, C. Burges, and A. Smola, eds., MIT Press, (1999).
15. Pratt, W. Dynamic organization of search results using the umls. In *American Medical Informatics Association Fall Symposium*, 1997.
16. Pratt, W., Hearst, M. and Fagan, L. A knowledge-based approach to organizing retrieved documents. In *Proceedings of AAAI-99.*
17. Shneiderman, B., Feldman, D. and Rose, A. Visualizing digital library search results with categorical and hierarchical axes. CS-TR-3993, UMIACS-TR-99-12. ftp://ftp.cs.umd.edu/pub/hcil/Reports-Abstracts-Bibliography/99-03html/99-03.html
18. Wittenburg, K. and Sigman, E. Integration of browsing, searching and filtering in an applet for information access. In *Proceedings of ACM CHI97: Human Factors in Computing Systems,* (Atlanta GA, March 1997).
19. Zamir, O., and Etzioni, O. Grouper: A dynamic clustering interface to web search results. In *Proceedings of WWW8* (Toronto, Canada, May 1999).
20. Zamir, O., and Etzioni, O. Web document clustering: a feasibility demonstration. In *Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*, 46-54.
21. http://cha-cha.berkeley.edu/
22. http://search.msn.com/
23. http://www.inktomi.com/new/press/directory.html/
24. http://www.looksmart.com/
25. http://www.northernlight.com/
26. http://www.snap.com/
27. http://www.yahoo.com/