An Experiment in Enhancing Information Access by Natural Language Processing*

Isaac Cheng and Robert Wilensky
Division of Computer Science
University of California, Berkeley

* This work was supported as part of the NSF/NASA/DARPA Digital Library Initiative, under NSF IRI 94-11334.

Contents

ABSTRACT **IV**

1. MOTIVATION **1**

2. DISAMBIGUATION AND TOPIC ASSIGNMENT **2**

2.1 THE LEXICAL DISAMBIGUATION ALGORITHM **2**
2.1.1 SOME ALGORITHMIC DETAILS 5
2.1.2 EFFICIENCY AND IMPLEMENTATION DETAILS 6
2.2 THE CLASSIFICATION ALGORITHM. **6**
2.3 PREPROCESSING **8**
2.3.1 DOCUMENT PREPROCESSING 8
2.3.1.1 Stemming. 9
2.3.2 THESAURUS PREPROCESSING 10

3. SOFTWARE DESIGN OF IAGO! **11**

3.1 OVERALL DESIGN **11**
3.2 THE INTERNET DIRECTORY **12**
3.2.1 THE DIRECTORY USER-INTERFACE 12
3.3 SEARCH BY WORD SENSES **14**
3.3.1 LEXICAL DISAMBIGUATION FILTER 14

4. EXPERIMENTS **16**

4.1 IAGO! 0.1 **16**
4.1.1 MOTIVATIONS FOR IAGO! 0.1 17
4.1.2 INITIAL RESULTS AND REMEDIAL MEASURES 18
4.2 EVALUATING THE INTERNET DIRECTORY **19**
4.2.1 METHODOLOGY 19
4.2.2 RESULTS 21
4.3 EVALUATING DISAMBIGUATION **21**

4.3.1 METHODOLOGY 21
4.3.2 RESULTS 22
4.3.3 EVALUATING SEARCH BY WORD SENSES 26

5. LIMITATIONS AND POSSIBLE IMPROVEMENTS 27

5.1 EFFICIENCY 27
5.2 IMPROVING DISAMBIGUATION 28
5.2.1 LIMITS OF THE APPROACH 28
5.2.2 THESAURAL CATEGORIES AS WORD SENSE PROXIES 29
5.2.3 INCOMPLETENESS OF THE THESAURUS 30
5.2.4 MULTI-WORD PHRASES 31
5.2.5 WORD ELEMENTS 32
5.2.6 USING THE WORD SENSE DISTRIBUTION TO IMPROVE DISAMBIGUATION 32
5.3 TOPIC ASSIGNMENT 32
5.3.1 THESAURAL CATEGORIES AS TOPICS 32
5.3.2 MULTIPLE CATEGORIZATION AND RANKING 33
5.3.3 DISAMBIGUATION 33
5.3.4 COMMON WORDS 34
5.3.5 MULTILINGUAL CONSIDERATIONS 34

6. EXTENSIONS AND OTHER APPLICATIONS: 35

6.1 AUTOMATED SUMMARIZATION 35
6.2 QUALITY 36
6.3 INTEGRATING INFORMATION FROM PICTURES 36
6.4 QUERY EXPANSION 37

7. CONCLUSION 37

8. ACKNOWLEDGMENTS 38

9. REFERENCES 39

An Experiment in Enhancing Information Access by Natural Language Processing

Isaac Cheng and Robert Wilensky

Abstract

We explore the hypothesis that lexical disambiguation could be applied to provide useful information access services. Specifically, we refined a lexical disambiguation method, and used it in a fully automatic categorization algorithm we developed. We also used this method more directly, to implement a service that retrieves documents by word sense.

To test these algorithms, we developed an experimental system, IAGO![1], in which we applied these algorithms to accessing the World Wide Web. IAGO! comprises both an Web directory (i.e., a classification of articles by topic) and a Web search service. Unlike most other Web directories, IAGO!'s directory was generated by a fully automatic process. One experiment shows a cataloging accuracy of 97%.

[1] IAGO /jiˈjɑːgəʊ/ stands for Isaac's Automatically Generated Ontology.

To improve net searching, IAGO! enables users to refine their queries by first detecting lexical ambiguities, and then allowing users to select specific word senses by which to search. IAGO! returns only Web pages in which a given keyword occurs in the specified sense. To help evaluate these results, we derive some performance thresholds that a disambiguation algorithm needs to operate above in order to be useful for retrieval. Our experimental results suggest that the implemented algorithm is performing well above these needs.

Keywords: natural language processing, World Wide Web, Web, Internet, Net, Intranet, Extranet, client-server, thin client, artificial intelligence, computational linguistics, word sense, sense, digital libraries, information systems, information technology, information retrieval, text retrieval, text searching, net search, search, index, text classification, Internet directory, Yahoo, IAGO!, business directory, yellow pages, multimedia, MM, Internet marketing, marketing, advertising, ad, commercial, business, entertainment, IS, MIS, IT

1. Motivation

Most information services today, such as those commonly available for the World Wide Web, provide access to documents via keyword indexing of various sorts, or by providing hand-generated directories. Such services leave much to be desired, both in terms of precision (i.e., relevance of the resulting documents) and recall (completeness of the result). Using natural language processing (NLP) to improve this situation is of course appealing: Ideally, one would like to express one's information need in natural language, and be presented with a result that is human-like in its precision and machine-like in its recall.

While its potential seems compelling, in practice, it is not clear that NLP has been of much help: Part of the problem is that natural language understanding is an "AI-complete" problem. To perform the sorts of semantic analysis necessary for most interesting natural language tasks requires poorly understood inference mechanisms and representations for common sense knowledge as well as knowledge of a given domain, not to mention an extensive lexicon, grammar, parser and so forth. Much progress needs to be made before robust, domain-independent, and scalable systems will be deployed that demonstrate mastery of significant language skills.

While "full" natural language processing is some distance away, we ask the question of which sub-problems might be addressed so as to offer more immediate, if more limited, possibilities. Of the many aspects of natural language understanding that one might try to exploit to help facilitate information access, lexical disambiguation seemed to us to offer the greatest potential. Some arguments to this effect are presented in [14]. While one may regard the experimental results to be inconclusive, we take heart in the finding that the effectiveness of disambiguation seems proportional to the shortness of a query, as short queries seem to be the norm (accounting for 40% of the queries on the World Wide Web [21]). Intuitively, this makes sense: The longer the query, the greater the chance that the terms will in effect disambiguate each other. With a single ambiguous term, we are guaranteed a poor result. Of course, in such a

situation, we cannot automatically disambiguation the query term either, but it seems reasonable to request the user to select a word sense in such situations. Therefore, a "query refinement" model, in which users interactively specify word senses, which are automatically matched against imputed senses in documents, seems to be a plausible paradigm.

While we can demonstrate individual cases of lexical disambiguation that require essentially arbitrary world knowledge, recent progress in statistical natural language processing suggests that it might be possible to solve this problem sufficiently well to facilitate the imperfect process of information retrieval. In particular, some methods for lexical disambiguation, especially [9], [17] and [23], seem promising enough to attempt to use over large and diverse collections. In addition, work ([6], [12] and [18]) has shown that related techniques could be used to perform automatic categorization as well. Automatic categorization is especially interesting for a collection like the Web, which is large and continually changing. Existing Web directories are created by hand, and hence, index only some quite small portion of the Web's contents. Automating this classification process could provide considerably enhanced value.

Therefore, it seemed to us that it was worthwhile attempting to see if large-scale disambiguation and automatic classification could be done. As an experiment, we attempted to build a more advanced form of Web service that offers searching and Web directories. In addition to searching by word, we would provide searching by word sense. We would provide a topical Web directory, but our directory would be generated fully automatically. IAGO! is the information access service that resulted from this experiment.

2. Disambiguation and Topic Assignment

2.1 The Lexical Disambiguation Algorithm

The core of IAGO! a lexical disambiguation algorithm. We chose as a basis the algorithm described by Yarowsky in [23]. This algorithm was appealing to us because it purports to require no manual intervention, an important feature for a system that needs to apply to any number of word senses of unrestricted text.

Yarowsky's algorithm, like [9], uses statistical correlations between word senses and words in a surrounding context to predict the sense of a word in a given context. However, rather than trying to learn the associations of word senses with nearby words, the algorithm learns associations of thesaural categories with nearby words, and uses the prediction of the category to determine a word sense. Thus we only have to learn the associations of vocabulary items and categories, a much smaller number than the number of the associations of word senses with vocabulary items. More importantly, the algorithm avoids the need for hand-tagging word senses by using the appearance of a word in a category of a thesaurus as a proxy for a word sense.

For example, suppose the word "bank" occurs in the Finance and the Land categories of one's thesaurus. We interpret this fact as there being two senses of "bank", one related to Finance and the other to Land. If we could predict which category were pertinent to a use of "bank" in a given sentence, we would have disambiguated that occurrence of "bank". We can make such a prediction by exploiting data acquired during a training phase, in which we identify the correlation of terms with categories. For example, during the training phase, we might have noticed that words like "money", "loan", "sale", etc., tend to co-occur near words that have a Finance sense. Of course, such words would also occur near words that have other senses as well. Moreover, the words co-occurring around these terms are likely to be ambiguous, and, without a corpus tagged for word senses, we won't know for sure which category a word is co-occurring with. For example, "sale" might occur near the term "check", which has an Inquiry sense as well as a Finance sense. However, over a sufficient quantity of training text, we can expect that the correlation of "sale", etc. with Finance will be strengthened by its co-occurrence with other terms in the Finance category, but that its correlation with Inquiry will not be.

The result of training is a matrix of associations of words with categories. We can think of this matrix as a set of vectors, one for each word, encoding the association of that word with each thesaural category. When the algorithm is deployed, the vectors of words surrounding a target word suggest the categories each word has been found to be associated with, and this evidence is combined to hypothesize the category of the target word, i.e., to disambiguate it.

That is, to disambiguate a given word, we would like to know *p(category | context),* i.e., the probability that a thesaural category occurs in a particular context, for each category of which that word is a member, and then select the maximum such value. This is equivalent to $p(category)\frac{p(context \mid category)}{p(context)}$, which, if we assume independence, becomes $p(category)\prod_{w_i \in context}\frac{p(w_i \mid category)}{p(w_i)}$, where w_i is a word in the context. The training phrase estimates each of the quantities $p(category)$, $p(w_i \mid category)$, and $p(w_i)$.

More specifically, the algorithm consists of two phases. During the training phase, the algorithm computes the frequency of co-occurrence of each word type with each category. It does so by establishing a moving window of 50 words around a target word, and associating each category to which the target word belongs with each word in the surrounding window. In Yarowsky's algorithm, each category appears to have been counted once, as if the sequence of 101 words occurred once for each sense of the word. In our variation, we normalized the occurrence of each category by the number of senses of that word. In effect, we make the assumption that the word senses of each word have a uniform distribution. (Later on, we will attempt to learn the actual distribution.)

In the deployment phase, the algorithm looks up the categories in which a potentially ambiguous word appears in the thesaurus. Each word in the context of the ambiguous word "votes" its category associations learned in the training phase. By adding up the votes the algorithm decides the most likely category and claims it as the sense of the ambiguous word.

Figure 1 and Figure 2 show an outline of the algorithm.

```
for each word w in a corpus,
        senses ← the thesaurus categories in which w is listed
        n ← number of senses
        for each category c in senses,
                g(c) ← g(c) + 1 / (n f(w))
                context ← 50 words that precede w and 50 words that follow w*
                for each word t in context,
                        A(t, c) ← A(t, c) + 1 / (n f(w))
```

The matrix **A** represents the co-occurrence frequency between each word and each category. f(*word*) is the frequency of the word (computed before training). g(*category*) is the estimated frequency of the category.

Figure 1: Disambiguation Algorithm: Training Phase

```
for each word w in a document,
        senses ← the thesaurus categories in which w is listed
        for each category c in senses,
                context ← 50 words that precede w and 50 words that follow w*
                for each word t in context,
                        salience ← A(t, c) / (f(t) g(c))
                        evidence ← log(salience)
                        Votes[c] ← Votes[c] + evidence
        the sense of w ← arg max Votes
```

Figure 2: Disambiguation Algorithm: Deployment Phase

2.1.1 Some Algorithmic Details

We suspect that such algorithms have not been widely used because of the difficulty of obtaining a reasonable quality on-line resources. For example, Hearst had to create a thesaurus automatically from WordNet. That thesaurus was used in [6] and [12], but was not of sufficient quality for the results to be more than suggestive. Our own several attempts to create a better thesaurus either automatically or with a relatively modest degree of user intervention were not notably successful. Eventually, we secured from HarperCollins an electronic copy of Roget's Fifth Edition [1], on which the results described herein are based. The

* In the cases where there are less than 50 words on either side of the ambiguous word, the context window was shrunk in those cases. For example, the first word of a document has a 51-word window, the second word has a 52-word window, and so on.

experiments described in [23] were performed using Roget's Fourth, so our results are not directly comparable, although in most cases we believe they would be quite similar. On the whole, Roget's Fifth is appears to be of higher quality and better suited for the task at hand. (However, this is not uniformly the case. See Section 5.2.3)

Yarowsky smoothed his observed word-category associations taking into account the probability of the word, using the algorithm described in [8]. We implemented this algorithm, but found that the simple maximum-likelihood estimate produced better accuracy (and ran much faster), and so we ultimately use this approach instead. We can only speculate about why this is the case.

Note that this algorithm was intended only for nouns, a limitation we did not attempt to overcome.

2.1.2 Efficiency and Implementation Details

For efficiency, the semantic locality in natural-language discourse was mapped into spatial locality. The word-category co-occurrence frequency matrix was stored as a B+ tree. Words are mapped to integers according to the order that they are listed in the thesaurus. Consequently, synonyms are adjacent to each other, and related words are close to one another. In a well-written document, in which the topics are localized, the storage access pattern is also likely to be localized.

The algorithm was implemented in C. A training session using about 10 million words took 32.66 hours on a Sun Ultra SparcStation utilizing 100 MB of physical memory. The co-occurrence frequency matrix occupies 393 MB of disk space. The system took 76.75 hours to disambiguate about 10 million words on the same machine.

2.2 The Classification Algorithm.

There has been considerable work on automatic text classification, ranging from applying conventional information retrieval techniques [15] to hand-crafted rules [11] to clustering [4]. We will not attempt to survey these systematically here (although [16] provides a useful collection of approaches). Instead, we note that

there is an intuitive relationship between lexical disambiguation and topic. [18] exploited semantic features from a machine-readable dictionary in just this spirit. Moreover, given that a disambiguation algorithm of the sort we have described is available, it should be possible to perform fully automatic categorization into the thesaural categories, without writing any rules or establishing any hand-tagged training sets. [12] used the associational vectors from Yarowsky's algorithm for each word in a text to suggest a thesaural category as a topic. The approach appeared promising, although the granularity of category assignment was limited (to ~100 categories).

Fisher [6] compared several families of algorithms based on disambiguation. Three families of algorithms survived an initial experiment and were subjected to more careful measurement. These were as follows:

(i) As in [12], add up the associational vectors, computed by the training phase of [23], for all the words in a text.

(ii) Use the disambiguation algorithm to establish the frequency of word senses in a corpus (the word sense "priors"[2]), and then assign topics by weighting each word's contribution to the categories to which it is a member in proportion to the word sense distribution in the corpus. For instance, if "bank" referred to a financial institution 90% of the time and a side of a river 10% of the time, it would cast a 0.9 vote to the Finance category and a 0.1 vote to the Land category.

(iii) Disambiguate the words of the text, and count only their imputed sense where possible; weight by sense priors (i.e., method (ii)) otherwise.

From his experiments, Fisher concluded that algorithms that exploited the word senses performed better than those that used only word-category association (although he was unable to detect a significant difference between methods (ii) and (iii)). The result is consistent with the intuition that word senses provide more

[2] The term "prior" is used here to mean **P**(category | word) in order to be consistent with [6]. Notice that it is a conditional probability and not to be confused with **P**(category).

direct evidence of the main content of documents than pure word-category associations.

In our experiments, we used method (ii) because, like Fisher, we have the associational vectors available, and because using word sense priors is much more efficient than performing disambiguation on the fly, without apparently much loss of accuracy. That is, we first use the disambiguator described above to estimate the word sense distribution of the words in a 10,000,000 word sample. To assign a text to a topic, the automatic topic assignment algorithm computes a vector $\boldsymbol{x} = (x_1, x_2, ..., x_{1073})$ (one element for each of Roget's) by summing the word sense distributions of each element in the text. The classifier outputs the index of largest component, $c = arg\ max\{x_1, x_2, ..., x_{1073}\}$, as the category for that document.

On the negative side, this algorithm will only classify text into the categories given by the thesaurus, which are sometimes of questionable utility. In addition, we have not yet examined the possibility that variant (iii) might be superior to (ii), which is certainly possible, given the higher quality of our thesaurus and training material compared to what as available to Fisher. Finally, like Fisher, we use the disambiguation algorithm to compute the distribution of word senses. The disambiguation algorithm itself is only correct some percentage of the time, so these "priors" are only estimated.

Using this algorithm, it took 20.19 hours to classify 18,614 Web pages on a Sun Ultra SparcStation.

2.3 Preprocessing

We performable a considerable amount of preprocessing of documents for disambiguation and for topic assignment; we also preprocess the thesaurus.

2.3.1 Document Preprocessing

The document preprocessor takes documents as input and turns them into a format that is used for classification and disambiguation.

- Convert HTML tags for parts-of-speech tagging. This is the only step of pre-processing that is specific to the document format details. Mostly, HTML tags (including comments) are simply removed. Paragraph markers are translated into a format recognizable to the part-of-speech tagger, and Javascript elements are eliminated altogether.
- Determine the part of speech of each word using a stochastic parts-of-speech tagger [3].
- Remove proper nouns. We eliminate proper nouns because of their uneven coverage in the thesaurus. This problem is discussed further in 4.1.2 and 5.2.3
- Remove punctuation marks.
- Remove common words using a stop-list.
- Convert parts-of-speech tags into a format understood by WordNet [19]. We use our own stemmer, which is WordNet-based, hence this step.
- Stem each word. We used our own stemmer, for the reasons described below.
- Remove documents that do not have enough words in them. Documents require a sufficient number of English words for the algorithms to apply to them. See sections 4.1.1 and 5.3.5 for further discussion.
- Map each word into an integral index.

2.3.1.1 Stemming.

As in most retrieval systems, we required the use of a stemmer. However, as discussed in [14] and [17], for tasks such as this one, it is desirable to remove inflectional morphemes but not derivational ones. For example, while the stemmer should reduce “apples” to “apple”, it should not reduce “glasses” to “glass” because otherwise the potential (and probably correct) “spectacles” sense of “glasses” would be lost, while the (highly unlikely) “transparent material” sense would be introduced.

As the more commonly available stemmers are overzealous in this regard, we developed our stemmer by modifying the stemmer in WordNet to use Roget’s Thesaurus as its vocabulary. I.e., when a term is listed explicitly in Roget’s, no

stemming is performed; otherwise, the stemmer attempts to remove inflectional morphemes.

Figure 3 contrasts the performance of the IAGO! stemmer with that of the SMART retrieval system, and with an implementation of Porter stemming algorithm, on a number of representative examples.

The SMART stemmer

```
% tstem ate
ate
% tstem apples
appl
% tstem formulae
formul
% tstem appendices
appendix
% tstem implementation
imple
% tstem glasses
glass
%
```

The Porter stemmer

```
% pstemmer ate
at
% pstemmer apples
appl
% pstemmer formulae
formula
% pstemmer appendices
appendic
% pstemmer implementation
implement
% pstemmer glasses
glass
%
```

The IAGO! stemmer

```
% stem
ate|2
eat|2
apples|1
apple|1
formulae|1
formula|1
appendices|1
appendix|1
implementation|1
implementation|1
glasses|1
glasses|1
%
```

Figure 3: Comparison of Stemmers

Unfortunately, the resulting stemmer incurs a hefty start-up time penalty (11 seconds on a Sun Ultra SparcStation) because of expensive initialization procedures that involve both WordNet and Roget's. Without counting the start-up time, the IAGO! stemmer takes 1.2 ms to stem a word.[3] We have not made any attempt to optimize the stemmer, which we believe to be a major readily eliminatable source of inefficiency in our system.

2.3.2 Thesaurus Preprocessing

We modified the source of Roget's in several ways. The most significant is the treatment of multi-word entries. While it would be desirable to include these, doing so generally would require some parsing of thesaural phrases. Our

[3] Performance evaluation was measured using 3000 words from Encarta 97. The stemmer utilized 164% of CPU time of a Sun Ultra SparcStation according to GNU time version 1.6.

expeditious solution to this problem was to eliminate multi-word phrases from the thesaurus altogether. This problem is discussed further in Section 5.2.4

The other modifications to the thesaurus consisted of removing or translating the internal markup format.

3. Software Design of IAGO!

3.1 Overall Design

To experiment with the ideas and algorithms described above, we developed an experimental system, IAGO!. IAGO! consists of two parts: a directory and a searching part (see Figure 4: Software Architecture of IAGO! 1.0).

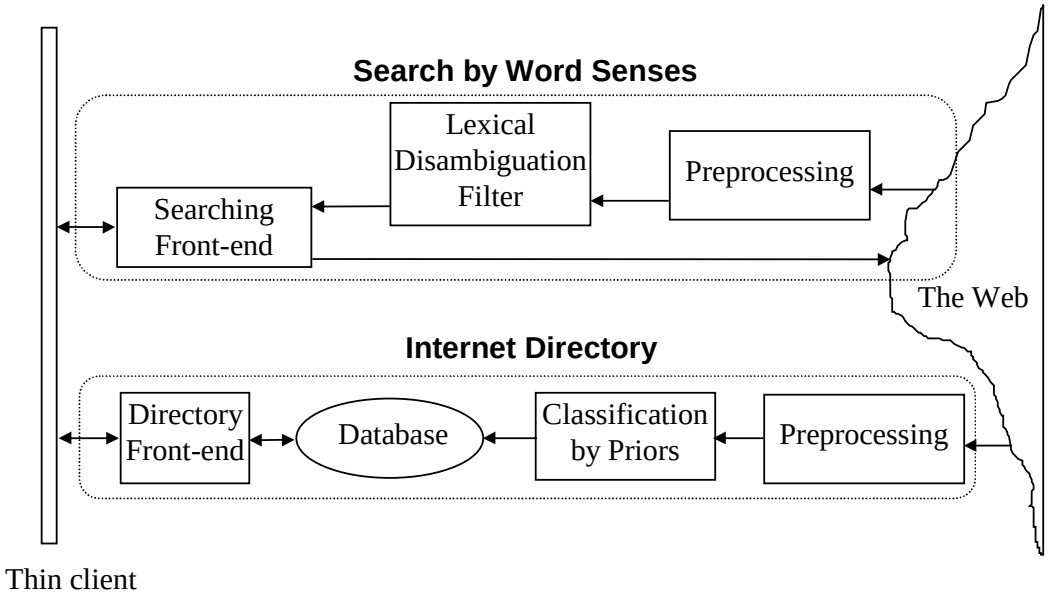

Figure 4: Software Architecture of IAGO! 1.0

A simple thin-client software architecture is adopted so that all the important processes run on the server. The only assumption on the client is that it

runs an HTML Web browser. The prototype was implemented mostly in Perl as a set of CGI scripts.

3.2 The Internet Directory

An Internet directory is generated as follows. The system obtains HTML documents from an Internet search engine [13]. The documents are pre-processed, as described above, and submitted to the classifier. The classifier produces a numerical value indicating the degree of relevance of the document to each category in the thesaurus. In the current implementation, the document is assigned only to the highest ranking category. The results of the classification are stored in a relational database.

3.2.1 The Directory User-Interface

A simple user interface, implemented as a CGI script, allows the user to navigate through a hierarchy (taken from the thesaurus), and ultimately sends a query to the database and returns the results to the user.

For example, if a user is interested in finding documents about animals, he or she may follow the steps illustrated by Figure 5.

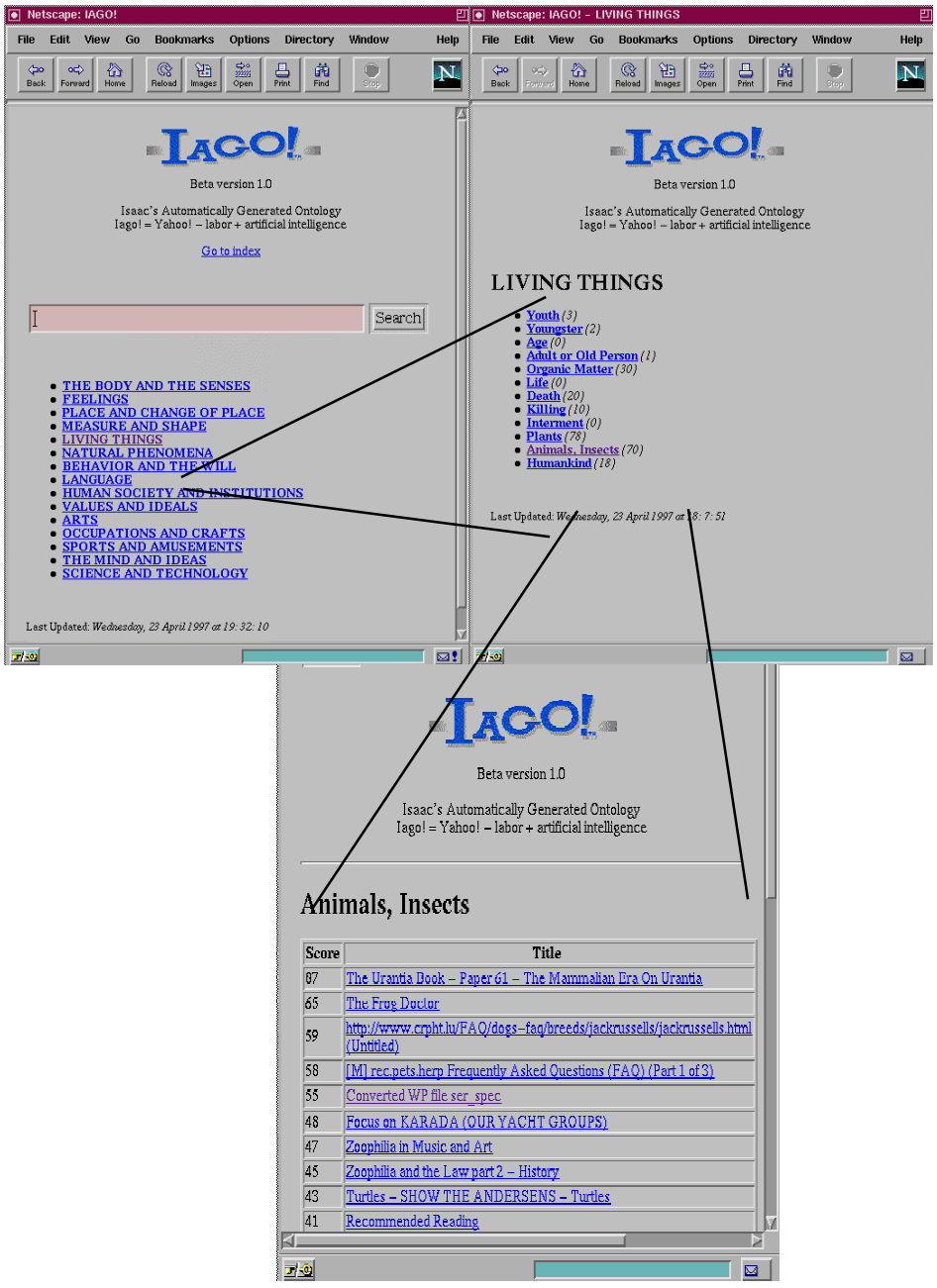

Figure 5: Internet Directory

In this example, the user selected “LIVING THINGS.” The system looked up the synopsis of categories of the thesaurus and displayed a list of categories pertaining to living things, from which the user chose the “Animals, Insects.” Finally, the system retrieved the URLs of the Web pages in that category from a database and presented the results to the user.

3.3 Search by Word Senses

3.3.1 Lexical Disambiguation Filter

The lexical disambiguation filter takes documents that match a keyword query as input, and filters out documents that do not match the senses that a user has specified. When a user has submitted a keyword query, IAGO! determines the ambiguity of the query by looking up the index of the thesaurus. If the keyword has more than one meaning, IAGO! will prompt the user to choose the desired senses. The keyword query is sent to an Internet search engine, which is asked to return some fixed number of matching Web pages. The resulting pages are passed to the preprocessor, and then fed into the lexical disambiguation filter.

Figure 6 shows an example in which a user entered “rock” as the query. After the user clicked the “Search” button, IAGO! looked up “rock” the index of the thesaurus. Since the word is listed in multiple categories in the thesaurus, IAGO! presented the entries from the index to the user, prompting the user to select the desired senses of “rock”. In Figure 6(a), the “stone” sense was chosen, and in Figure 6(b), the user chose the “rock-and-roll” sense.

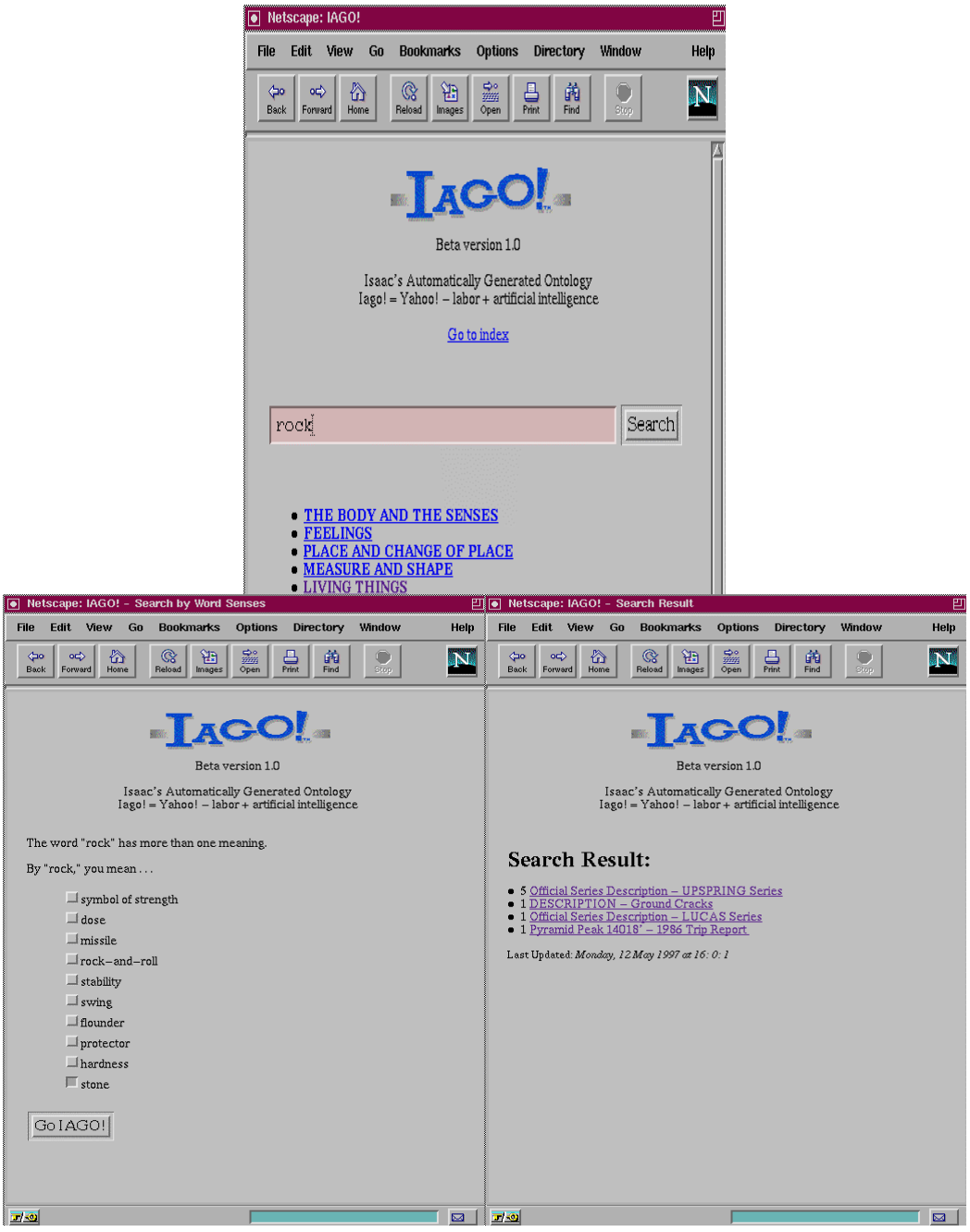

Figure 6(a): Search by Word Senses: "Rock" in the "Stone" Sense

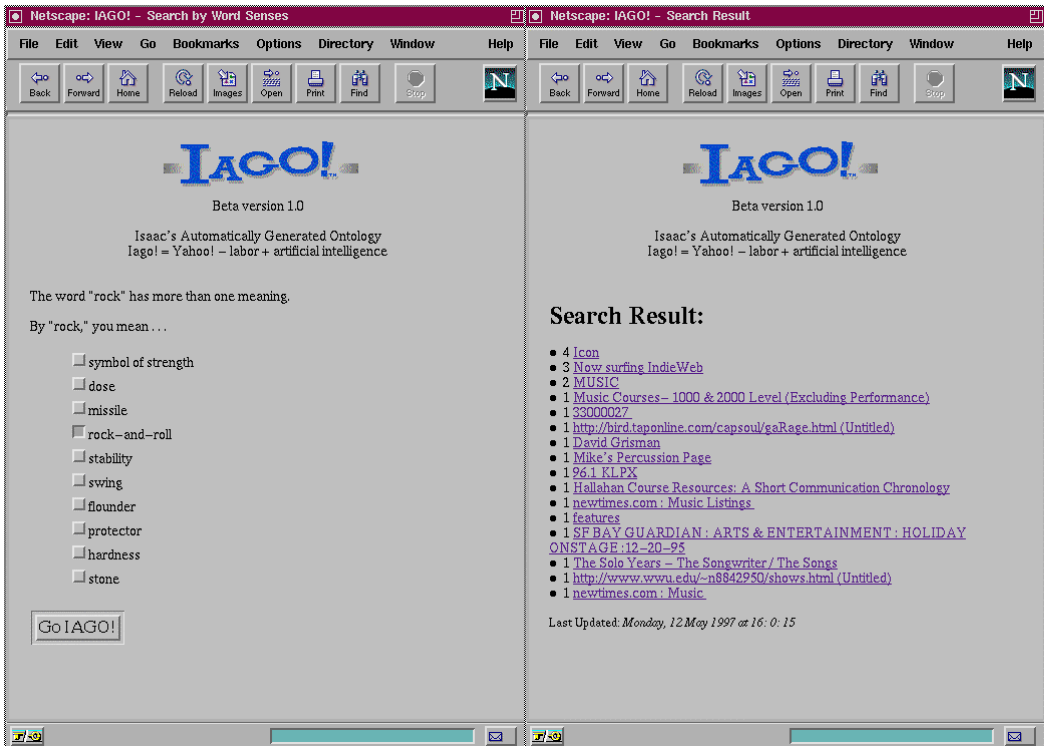

Figure 6(b): Search by Word Senses: "Rock" in the "Rock-and-Roll" Sense

4. Experiments

4.1 IAGO! 0.1

We began by constructing an initial version of IAGO!, which we will call IAGO! 0.1, both to test the initial feasibility of the idea, and against which to measure possible improvements. In this preliminary version, we filter out short articles, and both the training and the collection of the prior probability distribution were done using 10 million words from AP Newswire stories.

4.1.1 Motivations for IAGO! 0.1

We conducted an experiment with a preliminary version of the classifier to test our hypothesis that our algorithm would not reliably categorize short articles. To do so, we contrasted forcing the system to classify every Web page with classifying only those pages that had more than 100 content words. We classified Web pages using both procedures, and measured the correctness of the classification of the first 20 Web pages in each of the following categories: Computer Science; Finance/Investment; Tobacco; and Animals/Insects. Table 1 shows the percent of the first 20 documents that were correctly classified in this experiment.

	All Web pages	**Long enough Web pages only**
Computer Science	75%	100%
Finance/Investment	80%	100%
Tobacco	0%	18%
Animals/Insects	0%	60%

Table 1: IAGO! works better on Web pages that have enough text.

The precise percentages in each category are not significant because of the small size of the sample in this experiment. However, the large differences between the two columns strongly suggests that the classification algorithm does indeed do a better job of classifying Web pages that have a threshold amount of text in them. (I.e., since all the documents that are classified correctly in the "long enough" procedure will also be classified correctly in the "all pages" procedure, the reduction in accuracy in the latter case is due to enough shorter documents being incorrectly classified to preclude these correctly classified pages from appearing among the top 20 documents.)

Filtering out small pages will produce a reduction in the percent of the Web we would be able to classify. More accurately classifying a smaller percentage of the Web seems appropriate for this task, as the competition is human classification. In addition, we speculate, there may ultimately be other, the independent motivations for eliminating shorter Web pages. For example, it maybe that, on the whole, an independent valuation measure would tend to find very short pages to be of less interest than longer ones.

We trained on newswire stories at first because this was the text we had available to us. (Recall that our classifier exploits word sense distributions, obtained in a training phase, and that the disambiguator exploits associational information, also obtained in a training phase.) While it is not hard to obtain a subset of the Web to impute word sense distributions, it is difficult to obtain a relatively coherent training set, such as an encyclopedia, which, it has been observed, is more appropriate for the initial lexical disambiguation training [23]. We thought that the use of newswire stories as training data would be good enough to be suggestive about whether the approach was plausible, and provide a useful test of the importance of the type of training data. (It also lowers the threshold to replication of our results by others.)

4.1.2 Initial Results and Remedial Measures

Impressionistically, our initial trial results for IAGO! 0.1 indicated a reasonable level of performance for classification, but too low a level of performance for retrieval-by-word-sense. (We give precise results below.) We proceeded to analyze these results and determine performance improvements.

Based on the examination of the initial IAGO! 0.1 results, we instituted the following modifications:

- The training set. We believed this to be the most important factor in limiting performance. Fortunately, we were able to obtain a copy of the source of Microsoft Encarta 97 encyclopedia. We used this in place of newswire text to train the disambiguation algorithm.
- Computing word sense priors. To obtain priors for the distribution of word senses on the Web, we ran our disambiguation algorithm on 10 million words from documents on the Web, rather than on newswire text. (These words came from pages that did not overlap with the 18,614 Web pages we subsequently classified.)
- Proper nouns. IAGO 0.1 performed some bizarre classifications. For example, many articles were falsely being classified as being about tobacco. The problem was that the term “Virginia” appeared in Roget’s only as a type of tobacco. This rather uneven coverage of proper noun uses was unfortunately the rule rather than the exception. For example, the thesaurus does not list “Spanish” as a language or as people, but only

as an architectural style. Another problem of proper nouns is that the words in trademarks are often irrelevant to the content of the documents in which they appear. For example, the home page of the food company Birds Eye does not talk about birds or vision. Web pages with many such trademarks usually misled IAGO! 0.1's classifier. (This problem is essentially the same as that reported in [17], in that proper nouns are being mistaken for common nouns.) As an expedient, we simply eliminated proper nouns. Other solutions that are more attractive, but more effort to implement, are discussed in 5.2.3.

- The stop-list. The stop-list was augmented with some common content words, which, while correctly classified, did not seem to contribute to topicality. For example, in Roget's Thesaurus, "percent" is listed (only) in the Mathematics category, and "software" (only) in the Computer Science category. Most of the Web pages that contain the word "percent" are not about mathematics, and many Web pages that contained "software" are not about computer science. As a quick fix, we put those words into the stop-list and hence ignore them. An alternative solution discussed in Section 5.3.3

We call the system that resulted from these modifications IAGO! version 1.0.

4.2 Evaluating the Internet Directory

4.2.1 Methodology

While a large enough pre-classified test set is desirable, classifying a large number of Web pages manually is a tedious chore. This dilemma was resolved by getting a test set from Yahoo!, which provides a manually created the Internet directory. To make use of Yahoo!'s classification of the Web, we created a partial correspondence of Roget's categories to those of Yahoo!. Specially, nine categories were chosen from Roget's Thesaurus that had reasonable correspondence to some categories in Yahoo!. The mapping of the categories is shown in Table 2.

Roget's categories	Yahoo!'s categories
Computer Science	http://www.yahoo.com/text/Business_and_Economy/Companies/Computers/Research/ http://www.yahoo.com/text/Computers_and_Internet/Communications_and_Networking/Routing_Technology/Research_Groups/ http://www.yahoo.com/text/Computers_and_Internet/Operating_Systems/Research/ http://www.yahoo.com/text/Computers_and_Internet/Software/Databases/Research_Groups/
Finance, Investment	http://www.yahoo.com/text/Business_and_Economy/Companies/Financial_Services/Investment_Services/ http://www.yahoo.com/text/Business_and_Economy/Markets_and_Investments/Personal_Finance/
Fitness, Exercise	http://www.yahoo.com/text/Business_and_Economy/Companies/Health/Fitness/ http://www.yahoo.com/text/Health/Fitness/ http://www.yahoo.com/text/Business_and_Economy/Companies/Health/Fitness/Health_Clubs/ http://www.yahoo.com/text/Health/Fitness/Bodybuilding/ http://www.yahoo.com/text/Recreation/Sports/Running/ http://www.yahoo.com/text/Recreation/Outdoors/Walking/
Motion Pictures	http://www.yahoo.com/text/Entertainment/Movies_and_Films/ http://www.yahoo.com/text/Business_and_Economy/Companies/Entertainment/Movies/
Music	http://www.yahoo.com/text/Business_and_Economy/Classifieds/Music/ http://www.yahoo.com/text/Entertainment/Music/Genres/Classical/ http://www.yahoo.com/text/Entertainment/Music/Genres/Classical/Symphony_Orchestras/ http://www.yahoo.com/text/Entertainment/Music/Genres/Rock/ http://www.yahoo.com/text/Entertainment/Music/Genres/Rock/Metal/
Nutrition	http://www.yahoo.com/text/Health/Nutrition/ http://www.yahoo.com/text/Business_and_Economy/Products_and_Services/Health/Nutrition/ http://www.yahoo.com/text/Business_and_Economy/Companies/Health/Nutrition/
Occupation	http://www.yahoo.com/text/Business_and_Economy/Employment/Jobs/ http://www.yahoo.com/text/Business_and_Economy/Employment/Careers/
The Environment	http://www.yahoo.com/text/Business_and_Economy/Companies/Environment/ http://www.yahoo.com/text/Business_and_Economy/Companies/Environment/Water/
Travel	http://www.yahoo.com/text/Business_and_Economy/Companies/Travel/ http://www.yahoo.com/text/Business_and_Economy/Companies/Health/Travel/ http://www.yahoo.com/text/Business_and_Economy/Companies/Newsletters/Travel/

Table 2: The mapping between Yahoo!'s categories and Roget's

We then extracted from Yahoo! about 1000 Web pages that had more than 32 content words in them. As discussed above, filtering out small Web pages should give us a more accurate classification of a smaller portion of the Web. We chose the number 32 as a compromise: About 56% of the Yahoo! pages we sampled survived this filter, versus 28% that contained a 100 or more content words. Using the latter probably would have given us better results on the smaller sample, but the sample would have effectively been cut in half.

The classification of Yahoo! was taken as ground truth, but in certain gray areas, benefit of the doubt was given to IAGO!. That is, we examined by hand those pages in which IAGO!'s categorization differed from Yahoo!'s. In some cases in which the results seemed reasonable, we counted the result as correct. For instance, a Web page that talked about hiking was classified into the "Fitness, Exercise" category by Yahoo!, but might be classified into the "Travel" category by IAGO!. In that case, IAGO! was not counted as wrong.

4.2.2 Results

The classifying power of IAGO! was measured using an extended notion of precision and recall. Again, for the purpose of evaluating a classification system, we define *precision* as the number of documents correctly classified into a category divided by the total number of documents (correctly or mistakenly) classified into that category; *recall* is defined as the number of documents correctly classified into a category divided by the number of documents that should be classified into that category. The precision and recall of the two versions of IAGO! are shown in Table 3.

Version 0.1

Category Name	Precision	Recall
ComputerScience	31.6%	17.1%
FinanceInvestment	94.4%	22.0%
FitnessExercise	100.0%	4.3%
MotionPictures	100.0%	57.1%
Music	97.5%	58.3%
Nutrition	80.3%	35.6%
Occupation	100.0%	13.1%
TheEnvironment	n/a	0.0%
Travel	50.0%	5.7%

Overall precision = **88%**

Overall recall = 23%

Version 1.0

Category Name	Precision	Recall
ComputerScience	87.5%	19.4%
FinanceInvestment	100.0%	13.4%
FitnessExercise	100.0%	1.8%
MotionPictures	100.0%	54.8%
Music	98.2%	42.4%
Nutrition	97.9%	29.9%
Occupation	97.8%	30.3%
TheEnvironment	n/a	0.0%
Travel	75.0%	15.4%

Overall precision = **97%**

Overall recall = 21%

Table 3: Classification accuracy of IAGO!

These are figures giving IAGO! the benefit of the doubt in those cases in which it produces a different, but, we think, plausible, classification. Without any adjustment of the raw numbers, the overall precision and recall for IAGO! 1.0 are 92.3% and 20.4%, respectively. Therefore, if we are subjectively biased toward our own system, this bias is not so great as to distort the qualitative significance of the results.

4.3 Evaluating Disambiguation

4.3.1 Methodology

We first tested the quality of the disambiguation algorithm, separate from its use as a Web search engine. For the ground truth of the experiment, we judged and hand-labeled the senses of 100 word tokens from a corpus of AP newswire text

for each of the following word types: “interest”, “issue”, “sentence”, and “star.” (On the average, there were 1.5 test word tokens per document.) Since Roget’s categories often gratuitously subdivide intuitive word senses into multiple categories, we grouped the categorical senses into more intuitive senses as shown in Table 4. The number of occurrences of each resulting word sense is given in parentheses.

Word	Sense	Roget’s category
“interest”	Curiosity (46)	Curiosity; Allurement; Event; Relation; Attention; Motivation; Importance; Prerogative; Undertaking; Selfishness; Injustice; Influence; Reasoning; Cause; Occupation; Use; Goodness; Association; Aid
	Finance (46)	Lending; Debt; Securities
	Share (8)	Apportionment; Property; Acquisition
“issue”	Topic (58)	Topic; Politics; Cause; Importance; Inquiry
	Periodical (9)	Book; Publication
	Stock (33)	Securities; Quantity
	Outcome (0)	Product; Solution; Effect; Posterity; Event; Relationship by Blood
	Escape (0)	Escape; Emergence
“sentence”	Punishment (96)	Legal Action; Condemnation; Judgment
	Syntactic unit (4)	Phrase; Wise Saying; Part; Speech
“star”	Space object (7)	The Universe/Astronomy; Rock
	Celebrity (82)	Superiority; Success; Skill; Repute; Honor; Motion Pictures; Entertainer; Importance; Goodness
	Star shaped symbol (0)	Insignia; Grammar

Table 4: Mapping Roget’s categories into senses

In the case of the word “star”, the part-of-speech tagger we used produced 11 errors. (It was correct for all the tokens of the other word types.) Since the disambiguation algorithm only operates on nouns, we eliminated the mis-categorized tokens from consideration. We ran the disambiguation algorithm on the remaining tokens; if it output a category that is a member of the tagged sense, it was marked as correct. Otherwise, it was marked as an error.

4.3.2 Results

We compared the disambiguation algorithms of IAGO 1.0 with IAGO 0.1 and with a baseline algorithm, which always picks the most common sense of a word.

We measured the accuracy of the algorithms, i.e., the percent of word occurrences correctly disambiguated. The results of the experiment are shown in Figure 7: Disambiguation Accuracy.

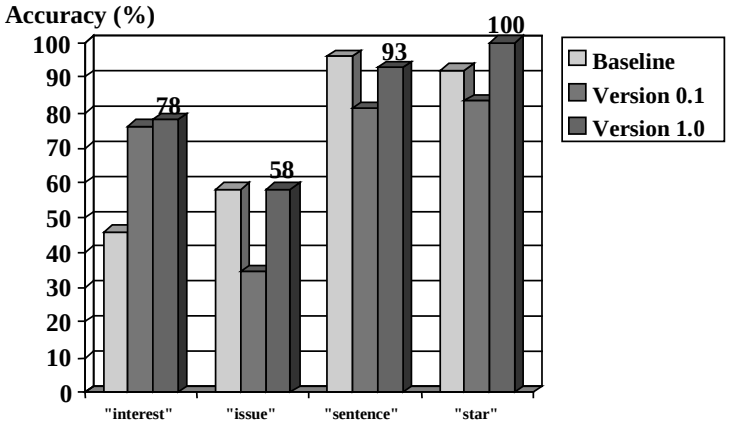

Figure 7: Disambiguation Accuracy

For this particular test suite, the difference between versions 0.1 and 1.0 is undoubtedly the value of training using Microsoft Encarta 97 encyclopedia. The results show that the quality of the disambiguation algorithm depends heavily on the training data. While this was what we anticipated, it is worth emphasizing that the improvement is due to a training set that is more coherent, not one that more closely resembles the target corpora. Moreover, the resulting improvement is the difference between the algorithm performing worse than the baseline and generally beating the baseline.

While comparison to such a baseline has been used elsewhere to evaluate the effectiveness of a disambiguation algorithm [9], the baseline understates the utility of a real disambiguation algorithm for retrieval. First, the most frequent sense of a term generally isn't known, so an algorithm exploiting the baseline may not be feasible to implement. However, even if the data were available, such a baseline-based algorithm cannot be used to make an effective retrieval algorithm, as the resulting algorithm could only return either all the documents containing a word, or none of them. If the resulting retrieval algorithm returns all documents

in which the word occurs when searching for the most frequent sense, and none when searching for a less frequent sense, then such algorithm would have no obvious utility.

Note further that a disambiguation algorithm that performs worse than the baseline is may still be useful for retrieval by word sense. While it is senseless to use the baseline method as a basis for retrieval by word sense, it may be reasonable to use a "real" method with the same, or even lower, accuracy. Assume a given word occurs in two senses, with *r* being the fraction of word instances in a corpus that are used in the more frequent sense. Assume for simplicity that the multiple word senses do not occur in the same document. Ordinary keyword retrieval is, in effect, equivalent to a disambiguation-based method that produces *r* precision and 100% recall when asked for the more frequent sense, and *(1-r)* precision and 100% recall when asked for the less frequent sense. A disambiguation method that was correct *p* of the time would have precision and recall values of $\frac{rp}{rp+(1-r)(1-p)}$ and *p*, respectively, for a word sense with frequently *r* in a corpus. Hence, an actual disambiguation algorithm operating at baseline efficiently would have precision and recall values of $\frac{r^2}{r^2+(1-r)^2}$ and *r*, respectively, when the goal is to find documents with the more common word sense, and .5 and *r*, respectively, when asked for the less common sense.

When *r* is large, these results compare favorably to those imputed for ordinary keyword retrieval, for retrieval by either word sense. Specifically, let us use the composite measure ***E***, defined as $1-\frac{1}{\alpha\frac{1}{precision}+(1-\alpha)\frac{1}{recall}}$, where $\alpha=\frac{1}{\beta^2+1}$, the parameter β being the preference of weighting towards recall or precision. Assuming a β of .5, a disambiguation method with comparable accuracy to the baseline is revealed as superior for retrieval by the more common word sense when *r* is larger than .57, and superior for retrieval by the less

common word sense when *r* is larger than .55. For example, with a 90/10 distribution of two word senses, a true disambiguation-based algorithm with a 90% accuracy would produce a 99/90 precision/recall for retrieval by the more frequent sense, and 50/90 for the less frequent sense, compared to 90/100 and 10/100, respectively, imputed for word sense retrieval obtained by using ordinary keyword search. ***E*** confirms the intuition that these are better overall results.

Indeed, a disambiguation algorithm can operate considerably below the "baseline" and still be useful. For example, again assuming a 90/10 distribution of word senses, then, for the more common word sense case, ***E***, with a β of .5, is better for a disambiguation algorithm with an accuracy over 77% than for keyword retrieval. (For the less common word sense, a "disambiguation" algorithm that is completely random gives a superior result.)

Consider now the cases in which *r* is small, i.e., when there are more senses per word, these having a relatively uniform distribution. In such a case, the disambiguation algorithm performing at baseline levels would be wrong most of the time, even if it is performing better than random, and hence considerably underperforms keyword search as a retrieval method. Let us use ***E*** to determine how good *p* has to be to make disambiguation worthwhile. As noted above, for a word sense of frequency *r*, and a disambiguation accuracy of *p*, precision and recall are $\frac{rp}{rp+(1-r)(1-p)}$ and *p*, respectively; using a keyword search as a proxy for finding the most common sense has the imputed precision and recall of *r* and 1, respectively. ***E*** is $1-\frac{1}{\alpha\frac{1}{precision}+(1-\alpha)\frac{1}{recall}}$, from which we can determine that, for the disambiguation algorithm to perform comparably to just doing a keyword search, *p* must be at least $\frac{\alpha-r(2\alpha-1)}{2\alpha-r(3\alpha-2)}$. Or, as *r* becomes small, *p* needs to be at least .5.

In other words, a disambiguation algorithm needs to be correct better than 50% of the time to be useful for retrieval, increasing in accuracy as the value of *r* increases beyond this point, although it can trail this value by some degree. As

the results above suggest, IAGO! 1.0 (and, for the most part, even IAGO! 0.1) performs well above the required figures.

In addition, we note that it is possible that the ranking given by a disambiguation algorithm may provide added value as well.

A major problem in evaluating our particular algorithm (and of the algorithm itself) is that the thesaural categories are not always a good representation of word senses. For example, out of the 42 errors in the above results on "issue", 18 errors was made by the algorithm assigning "issue" (in the topic sense) to the Event category. It is not completely clear whether the Event category should be regarded as the "topic" or the "outcome" sense of "issue." We subjectively chose the latter (see Table 4: Mapping Roget's categories into senses). If the former were chosen, the result would be 76% instead of 58%.

4.3.3 Evaluating Search by Word Senses

Since we have not pre-computed a word-sense index of the Web, IAGO! retrieves some number of Web pages using a keyword search engine, and then filter these for the desired word sense. Since, on the average, only a fraction of the pages that match the keyword will survive the filter, this number must be set large enough to produce some meaningful yield. On the other hand, processing time is proportion to the size of the keyword result set, and is substantial, as it requires waiting for the full text of the page to be returned by a server, and running our disambiguation algorithm on the result. After some experimentation, we determined that the underlying keyword search engine needed to retrieve about 500 pages, which is the number we used here. Note, though, that the Web crawler attempt to retrieve pages in batch, and times out after an interval, so that different sets of candidate documents are likely to be retrieved on repeated retrievals of the same word, depending on the vagaries of the network and server loads.

We evaluated several IAGO! retrieval-by-word-sense result sets by hand. The results appear quite promising. For instance, the query "'rock' in the 'stone' sense" returns four URLs; all four are correct. The query "'rock' in the 'rock-and-roll' sense" returns 16 URLs; all 16 were correct. I.e., IAGO! has 100% accuracy for both senses of "rock." The query "'chair' in the 'chairperson' sense returned

50 URLs, of which the top 10 were examined; only last one was incorrect. The query "'chair' in the 'seat' sense'' returned 26 URLs, of which the top 10 were examined; only the last two were wrong. These samples are consistent with the analysis performed above suggesting that disambiguation in IAGO! is operating well enough to be useful.

In addition, it appears to us that the documents ranked at the top (by the number of occurrences of matching word senses) are often the most relevant ones.

In many cases, though, the user must select a number of Roget's categories in order to submit the equivalent of a search by an intuitive word sense. For example, in our examples used above to test the disambiguation algorithm, we merge nineteen Roget's categories to get one intuitive sense of the word "interest", and three for each of the other senses of "interest". Requiring the user to make such a selection clearly limits usability[4]. We discuss this drawback in section 5.2.2.

5. Limitations and Possible Improvements

5.1 Efficiency

Efficiency enhancement would be necessary to convert IAGO! from an experimental prototype to a real system. Fortunately, there is room for much improvement in this regard. IAGO! 1.0 currently runs on a single-workstation. Classification takes about two seconds per Web page; search by word senses takes about eight minutes per query. However, the natural language processing algorithms are very simple and easily parallelizable. In particular, the classification processes of different documents are independent of each other;

[4] IAGO!'s retrieval-by-word-sense interface presents the user with only those categories for which the word is listed in Roget's index. Words may appear in additional categories in the thesaurus. For example, the "rock" and "grammar" senses of "star" given in Table 4 are not listed in Roget's index, even though "star" does indeed occur in these categories. Presumably, the thesaurus considers these unhelpful to list, and we follow their example. It was still necessary to include these senses during training, however.

documents can be disambiguated on different machines with no inter-machine communication

A large and avoidable performance hit is due to the stemmer. The performance of this module should be easy to improve. In addition, we suspect there are significant performance gains to be had by coding in C various portions of the algorithm that are stitched together now with scripting language code.

In performing search-by-word sense, we must retrieve the actual text of Web documents, and then filter them. About half the time is spent simply crawling the Web. Also, since a large percent of indexed documents are short, or ultimately are found to contain unrequested senses, this "retrieval and filtering" paradigm produces a rather low yield (hence the small numbers reported in the evaluation above). To make a practical system, one would prefer to create a word sense index. To do so, it would be desirable to have the disambiguation algorithm operate at or near web-crawler/indexer speeds. We are far from this point at the moment, but believe it is achievable just by determined performance tuning.

5.2 Improving Disambiguation

5.2.1 Limits of the Approach

The approach to lexical disambiguation we have used is limited in several respects. First, it is intended primarily for nouns. We have not tested it on other parts of speech.

The algorithm, like other statistical algorithms, is sometimes simply wrong. Consider the following example:

> *Copy desks were asked whether "gibe" or "jibe" was correct in the following* **sentence***: "Three suspects were taken to police headquarters but detectives announced later that their stories didn't gibe." The verdict was almost unanimous that "jibe" was correct.*

Here the algorithm picked the judicial sense instead of the grammatical sense of "sentence", presumably because of the confusing contexts words like "police", "detectives", and "verdict." We do not see any hope that a simple statistical algorithm can be made to correctly disambiguate such cases.

On the other hand, we note that there are some additional benefits to a purely statistical approach. Many documents in the Web contain useful terms that do not occur in recognizable sentences. For example, the terms may appear in tables or other structures that are generally not considered by more elaborate natural language processing algorithms. The purely associational nature of the disambiguation method applies directly and profitably to these cases.

5.2.2 Thesaural Categories as Word Sense Proxies

One of the major problems of the disambiguation algorithm used in IAGO! is that thesaurus categories are not always good representations for word senses. For example, the word "interest" appears in 25 of Roget's categories, giving us 25 ostensibly distinct senses. Generally, a number of such categories represent the same sense, in more or less direct ways. For retrieval-by-word-sense, the user can select multiple "senses" to approximate the intuitive sense, but this is a tiresome and unintuitive chore. In many cases, it is very difficult to guess which sense is intended by category membership.

[23] seems to have grouped categories together by hand to arrive a reasonable approximations to word senses; we did the same for the purposes of evaluation. However, it would be a large task to establish such grouping for each lexical item. We do not see a simple way to automatically establish the proper sense groups. Performing this task is an interesting research topic.

[17] used the Subject Field Codes from *Longman's Dictionary of Contemporary English* (LDOCE) for statistical word-sense disambiguation. Since these are labelings of actual word senses, they would no doubt be superior to using thesaural categories. The only reason we did not attempt to use these is incompleteness. According to [17], while 95% of the entries in LDOCE have Subject Codes, over half of these are the label "General", i.e., provide no semantic information. However, the algorithms we use are completely

independent of the source of the labelings. Therefore, the availability of such a set would provide an immediate large improvement in the quality and usability of these algorithms.

5.2.3 Incompleteness of the Thesaurus

In addition to the fragmentation of single senses among multiple categories, there is the problem of thesaurus gaps, in which a particular sense of a word is not listed in the thesaurus. Frequently, a word is missing entirely from the thesaurus. This is especially true for technical terms and abbreviations, which seem to enter the language rapidly. For example, the abbreviation "PC" is now widely used to mean both "personal computer" and "politically correct"[5]; "NOW" is both the "National Organization of Women" and "Network of Workstations". None of these senses are in our thesaurus.

The complete omission of words is not as damaging as omitted word senses: The omitted term contributes no evidence, but it does not mislead. The problem of proper nouns mentioned above is an example of the more pernicious problem of omitted senses. Fortunately, it is relatively straightforward to convert this problem into the lesser evil of omitted words simply by eliminating proper nouns from consideration altogether. A more edifying solution would be to merge in collections of proper nouns. Considerable work would be involved to enter these into the right thesaural categories.

Thesaural gaps exist for ordinary lexical items, where no immediate fix is evidence. For example, in Roget's Fifth, "crane" is not the Animal/Insects category, nor is "bass", so their respective bird and fish senses are missing. (Both gaps appear in the 5th edition of Roget's International Thesaurus but not the 4th edition, which contains useful lists of many animal types.[6])

[23] suggests an algorithm for filling thesaural gaps automatically, but only appears to have applied it in one case. In addition, it is unclear how to

---

[5] The later phrase is not a noun phrase, so it is not directly relevant to the algorithms being described.

[6] The 4th edition has other gaps. For instance, the card playing sense of "suit" is missing, as mention in [23].

implement the algorithm efficiently. Thus, whether such gaps can be filled automatically, or a human thesaurus builder aided in finding and correcting such gaps, still remains to be seen.

5.2.4 Multi-word Phrases

We have not made any effort to exploit multi-words terms in IAGO! Knowledge of phrases would of course be useful in non-compositional cases, in which a category or meaning might be available that would not otherwise be suggested at all. In other cases, knowledge of multi-word phrases would appear to be useful for disambiguation and topic assignment involving those term's constituents. For example, knowledge of the phrase "space station" would allow one to conclude that with high probability that "space" is occurring in the astronomical and not the typographic sense.

Since many phrases are listed in Roget's Thesaurus, IAGO! could be extended to exploit them. The primary problem with doing so is that it would entail parsing thesaurus entries. For instance, in Section 184.10 of the thesaurus is a phrase "nonscheduled airline *or* nonsked." It should be parsed into "nonscheduled airline" and "nonsked." On the other hand, in Section 184.31 is the entry "sonic barrier *or* wall", which should be parsed into "sonic barrier" and "sonic wall", not "sonic barrier" and "wall." In general, determining the intended parse of such phrases is not straightforward, although perhaps it would be possible to parse them with the aid of another phrasal knowledge source.

Using phrases to aid disambiguation is one of a large number of ways in which IAGO! might be extended to take additional types of context information into account. E.g., exploiting knowledge of the phrase "space station" is equivalent to saying that the occurrence of "station" immediately after the word "space" should be weighted differently from its occurrence anywhere within a window of 100 words. While doing so is intuitively appealing, testing this hypothesis would be a worthwhile experiment.

5.2.5 Word Elements

Bound morphemes are sometimes listed in Roget's. For example, "contra-" and "counter-" are listed in the Disagreement (788) category. IAGO! currently does not take advantage of such morphemes. Handling bound morphemes may be of some utility, but their number is probably too small to be significant.

5.2.6 Using the Word Sense Distribution to Improve Disambiguation

IAGO! 1.0 exploits the probability distribution we compute for word senses for topic assignment, but not for disambiguation. In effect, we make the assumption of uniform distribution of word senses at the initial training, and later compute a distribution of word senses in our corpus. It seems reasonable that the computed distribution should be helpful in improving disambiguation. That is, given *p(category), p(category | word)* (i.e. the computer sense distribution), and *p(category | context),* compute *p(category | word, context),* instead of just using *p(category | context).*

Given that *p(category | word, context)* is better than *p(category | context),* it should be possible to iterate as per various expectation-maximization algorithms, i.e., use the better disambiguator to improve the estimate of word sense priors. We can readily compute *p(category | word, context)* if the word and context are independent enough for simple Bayesian updating would work. We tried this approach in IAGO! 0.1 without success. We suspect that it failed because of strong dependencies between a word and its local context, but have not tested this hypothesis by attempting the experiment again with IAGO! 1.0. Performing such experiments may require further optimization of the system for efficiency. Currently, it takes a few days to train and a few days to collect priors.

5.3 Topic Assignment

5.3.1 Thesaural Categories as Topics

As with any other set of fixed vocabulary items, our specific results are limited to the utility of that set. Roget's is not without its peculiarities. Along with categories such as Materials, Mathematics and Minerals are Meaninglessness, Mediocrity, and Misteaching. In our particular implementation, having

undesirable categories is especially damaging, as we only assign a document to a single category, so that assigning a document to a low-utility category precludes a more useful assignment. Fortunately, the categories we intuitively regard as uninteresting are rarely assigned a documents. For example, in our experiment, Materials, Mathematics and Minerals were assigned 18, 564, and 19 Web pages; Meaninglessness, Mediocrity, and Misteaching were all assigned no pages. The category The Environment, which has the scientific interpretation, was assigned several articles; the category Environment, which has the abstract interpretation, was assigned no articles. And, of course, it would be a simple matter to put undesirable categories on a stoplist, and preclude their use altogether.

Roget's categories comprise a relatively high-level classification. This is useful as a general index, but not for finer-grain distinctions that might be useful within a specialty. (For example, it is certainly much coarser than the categories used by Yahoo!.) One way to produce more specific classifications would be to use a thesaurus specifically tailored to a particular domain. [2] describes a method for just such thesaurus construction. A useful experiment would be to apply the topic assignment algorithm described here to classify a collection from which such a thesaurus was generated.

5.3.2 Multiple Categorization and Ranking

IAGO! currently assigns at most one category per Web page. However, the underlying technology allows much more flexibility than the existing prototype offers. The automatic topic assignment engine outputs a topic vector $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ for each Web page, where n is the number of conceptual categories (n = 1073 for Roget's). The current classifier outputs the index of strongest component, $c = arg\ max\{x_1, x_2, ..., x_n\}$, as the category for the Web page, and its value, x_c, as the rank. It certainly seems plausible to find a principled way to use this output to assign multiple categories to a document, and, perhaps, to find better ranking criteria.

5.3.3 Disambiguation

The prior probability distribution of word senses is currently used for topic assignment to avoid the expensive disambiguation procedure during topic assignment. This was done only because of efficiency considerations. However,

we left an open question of whether using disambiguation in addition to priors can significantly improve topic assignment. With a faster implementation of the disambiguation algorithm and/or better hardware, one might explore the question empirically.

5.3.4 Common Words

Above we noted that some words do not seem to contribute to topicality. Examples are "percent" and "software", which belong to the Mathematics and Computer Science categories, respectively, but are arguably generally not evidence of the relevance of these categories. Terms naming numerical quantities and times also appear to have this property.

Our current solution to this problem is to put such terms on a stoplist. This approach worked very effectively.[7] However, it requires manual intervention. Rather than using a manually-constructed stoplist, an alternative might be to use the standard inverse document frequency (IDF) measure to generally attenuate the topicalization evidence of a word in proportion to the number of Web pages in which that word appears. Some experimentation would be required to assess the value of IDF for automatic topic assignment generally, and as an alternative to using a stoplist for problematic terms.

5.3.5 Multilingual Considerations

Pages on the Web are frequently contain languages other than English. IAGO! discards pages that do not have enough English terms in them, where "English term" is defined as being found in Roget's. On the other hand, IAGO! will attempt to categorize predominately non-English pages by their English contents, provided these are substantial enough.

---

[7] Putting such words on our stoplist does not interfere with retrieval by word sense, since the terms are unambiguous, and hence, IAGO! just passes them through to the underlying search engine. In the case of "software", though, no result is returned, as the term is on AltaVista's stoplist, presumably for a similar reason.

This approach seems to work well for languages such as Japanese and Chinese, which do not overlap much with English. (We have not measured how well IAGO! performs in these cases, but our impression, for the languages that we know, is that performance is good. Perhaps this is not surprising, as the heavy use of foreign words seems to occur in technical subjects, and hence these provide reasonable topic indications.)

European languages are more troublesome because they sometimes coincidently share a commonly used term with English. For example, in Swedish, the second person pronoun *man* ("you") and the third person pronoun *hon* ("she") are common terms. IAGO! would mistake these as the English words for a male adult person and the abbreviated form of the endearment "honey". Moreover, these would be common enough in Swedish to comprise a sufficient amount of "English" text in many documents for IAGO! to perform completely erroneous classifications. The Dutch terms "op" and "van" would cause similar difficulties.

IAGO! is prevented from making these errors because we put such problematic terms on the stop-list. A more principled solution may be to set up a separate foreign-language stop-list and ignore those terms only when they appear in a non-English page. A possible heuristic for detecting non-English pages is to examine the ratio of the number of words that pass through the preprocessing to the size of the Web page in bytes. A low ratio indicates that the page is probably not written in English.

6. Extensions and Other Applications:

6.1 Automated Summarization

In addition to classification and retrieval-by-word-sense, we suspect that the fundamental algorithms in IAGO! could also be used to generate summaries of Web pages. In particular, some systems now produce summaries by picking out sentences that are "close" to the topic of the document in which they appear. Closeness generally appears to be measured in "lexical space", in which each word type constitutes a dimension. This approach exploits a weak notion of

semantics, and involves dealing with a very large number of dimensions, essentially the number of words in the English vocabulary.

We suggest that IAGO! may overcome these drawbacks. The topic-assignment algorithm maps text into a 1073-dimensional vector, which can be used to compare documents with each sentence. It is possible that this method will introduce a somewhat richer notion of semantics. An experiment can easily be devised to test this hypothesis.

6.2 Quality

Services such as Yahoo! also filter for quality as well as topicality. One drawback of IAGO!'s automatic directory construction algorithm is that it makes no attempt to distinguish important documents from unimportant ones. Recent attempts at automatically making such distinctions seem promising, however. Such attempts generally use the structure of a Web region around a document to determine its importance, for example, the number of links pointing to a document. For example, [20] describes a technique that ranks pages on the web, using the link structure of the web to approximate the citation importance of a page. Filtering the Web first for importance would be probably be a large step in making IAGO! competitive with handmade directories.

6.3 Integrating Information from Pictures

IAGO! 1.0 understands only the text on Web pages. It ignores all pictures during the classification and searching processes. The technology of analyzing images automatically is still in too rudimentary a stage to help at present [7]. However, should we be able to hypothesize the objects in images with some reliability, one possibility is to substitute terms for images, perhaps repeating the term in proportion to the size of the image. Such terms could then be integrated together with the pre-existing text for the purposes of classification.

6.4 Query Expansion

Query expansion is an alternative means to interactively improving searches. Query expansion encourages users to lengthen their queries by picking additional keywords. As longer queries tend to disambiguate their constitute words, the two approaches might be viewed as competitive. Much would depend on the user's information need: Adding "guitar" to the query "rock" would help to disambiguate it somewhat, but may unduly limit the recall more than restricting word senses. Some more empirical study is required to better understand the relationship between the two techniques, and when each can be most productively exploited. Our suspicion is that the two approaches can be productively combined.

7. Conclusion

We believe these experiments demonstrate that automating directory generation and search by word senses are promising ways to enhance information access on large-scale information resources, such as the Internet. With only performance improvements, IAGO! would probably be a useful system now. We have demonstrated that, at the current point in time, lexical disambiguation can be done sufficiently well to be a useful addition to one's arsenal of information retrieval techniques. Combining IAGO! with other techniques, such as value filtering, would produce a system that may compete with hand-created document directories.

Many straightforward improvements are possible both to IAGO!'s algorithms as well as to the data sources it exploits. We improved performance by using a coherent training set; a better set of semantic codes in place of thesaural categories would likely yield very substantial additional improvements.

A number of avenues for further research and development have been suggested by this exercise. Based on our experience, we believe such research would be worth pursuing.

8. Acknowledgments

HarperCollins Publisher provided us with the full-text of Roget's International Thesaurus Fifth Edition. We are indebted to Ms. Carol Cohen of HarperCollins for her invaluable help in obtaining this resource. None of the work described herein could have occurred without it.

Microsoft Corporation contributed the full-text of Encarta 97. This training data was hugely importance to the success of this project. Microsoft was the only encyclopedia vendor that was reasonable to deal with. Special thanks to Bill Dolan for his persistent efforts not to take yes for an answer.

Professor Eric Brewer and Inktomi Corporation contributed the 2 GB of Web text that we used for computing sense distributions.

David Fisher helped us understand details of his previous work. Aitao Chen provided us with the source code of popular stemmers. Ginger Ogle and Joyce Gross gave us help on the Digital Library server. Ray Larson and Marti hearst provided us with useful comments on versions of this paper. Professor Richard Fateman, Professor Joe Hellerstein, Ginger Ogle, and Loretta Willis helped evaluate IAGO! 0.1 and gave us constructive comments.

Special thanks are due to Albert Tan and Timotius Tjahjadi. Albert built the B+-tree nodes and disk cache for storing the word-category association data, and worked closely with us in making significant efficiency enhancements. Tim implemented most of the front-end work of IAGO! 1.0 and a Web crawler for searching.

This research was funded as part of the NSF/NASA/DARPA Digital Library Initiative, under National Science Foundation grant number IRI-94-41286.

9. References

[1] Chapman, R., ed. *Roget's International Thesaurus*. 5th ed. HarperCollins, 1992.

[2] Chen, H. Yim, T., Fye, D., and Schatz, B. Automatic Thesaurus Generation for an Electronic Library Community System. In Jornal of the American Society for Information Science, 46(3); pp. 173-193, 1995.

[3] Church, K. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text." *Proceedings of the Second Conference on Applied Natural Language Processing*. Austin, Texas, 1989.

[4] Cutting, D., Karger, D., Pederson, J., and Tukey, J. W. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In the *Proceedings of the 15th Annual International ACM/SIGIR Conference*, Copenhagen, 1992.

[5] Fisher, D. and Riloff, E. "Applying Statistical Techniques to Small Corpora: Benefiting from a Limited Domain." *Proceedings of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, 1992.

[6] Fisher, D. "Topic Characterization of Full Length Texts Using Direct and Indirect Term Evidence." *Technical Report UCB/CSD 94-809*. Computer Science Division, University of California, Berkeley, May 1994.

[7] Forsyth, D. A., Malik, J., and Wilensky, R. Searching for Digital Pictures. *Scientific American*, June 1997.

[8] Gale, W., Church, K., and Yarowsky, D. "Discrimination Decisions for 100,000-Dimensional Spaces". AT&T Statistical Research Report No. 103, 1992.

[9] Gale, W., Church, K., and Yarowsky, D. "A Method for Disambiguating Word Senses in a Large Corpus." *Computers and the Humanities*, 5-6, pp. 415-439. 1992.

[10] Gale, W., Church, K., and Yarowsky, D. "Estimating Upper and Lower Bounds on the Performance of Word-Sense Disambiguation Programs." *Proceedings of the 30th meeting of the Association for Computational Linguistics*, pp. 249-256. 1992.

[11] Hayes, Philip J. and Weinstein, Steven P. CONSTRUE/TIS: A System for Content-Based Indexing of a Database of News Stories. In A. Rappaport and R. Smith, *Innovative Applications of Artificial Intelligence 2*, AAAI

Press/The MIT Press, 1990.

[12] Hearst, M. "Context and Structure in Automated Full-Text Information Access." Doctoral Dissertation. University of California, Berkeley, 1994.

[13] Inktomi Corporation. "The Inktomi Technology behind Hotbot: A White Paper." *http://www.inktomi.com/whitepap.html*, 1996.

[14] Krovetz, R. "Viewing Morphology as an Inference Process". In the Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 191-202, 1993.

[15] Larson, R. "Experiments in Automatic Library of Congress Classification." *Journal of the American Society for Information Science*, 43(2), pp. 130-148, 1992

[16] Lewis, David D. and Hayes. Philip J. (eds.) *ACM Transactions on Information Systems, Vol*. 12, *No*. 3, July 1994. Special Issue on Text Categorization.

[17] Liddy, E. D. and Paik, W. "Statistically-Guided Word Sense Disambiguation." *Proceedings of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, 1992.

[18] Liddy, E. D., Paik, W., and Yu, E. S. Text Categorization for Multiple Users Based on Semantic Features from a Machine-Readable Dictionary. In Lewis, David D. and Hayes. Philip J. (eds.) *ACM Transactions on Information Systems, Vol*. 12, *No*. 3, July 1994, pp. 278-295.

[19] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. "Introduction to WordNet: An On-line Lexical Data Base." *Journal of Lexicography*, vol. 3, no. 4, pp. 235-244, 1990.

[20] Page, L. http://www-pcd.stanford.edu/~page/papers/citeimport.html

[21] Pedersen, J. O. "Search: the Next Killer App?" Digital Information Systems Seminar, University of California, Berkeley, Oct. 14, 1996.

[22] Porter, M.F. "An Algorithm for Suffix Stripping". Program 14 (3), July 1980, pp. 130-137.

[23] Yarowsky, D. "Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora." *Proceedings of the Fourteenth International Conference on Computational Linguistics*, pp. 454-460, Nantes, France, 1992.