Abstract

Semantics-free, word-based information retrieval is thwarted by two complementary problems. First, search for relevant documents returns irrelevant items when all meanings of a search term are used, rather than just the meaning intended. This causes low precision. Second, relevant items are missed when they are indexed not under the actual search terms, but rather under related terms. This causes low recall. With semantics-free approaches there is generally no way to improve both precision and recall at the same time.

Word sense disambiguation during document indexing should improve precision. We have investigated using the massive WordNet semantic network for disambiguation during indexing. With the unconstrained text of the SMART retrieval environment, we have had to derive our own content description from the input text, given only part-of-speech tagging of the input.

We employ the notion of semantic distance between network nodes. Input text terms with multiple senses are disambiguated by finding the combination of senses from a set of contiguous terms which minimizes total pairwise distance between senses. Results so far have been encouraging. Improvement in disambiguation compared with chance is clear and consistent.

Keywords: Information retrieval, indexing, word sense disambiguation, semantic networks, free-text.

1 Introduction

Semantics-free, word-based information retrieval is thwarted by two complementary problems. First, search for relevant documents returns irrelevant items when all meanings of a search term are used, rather than just the meaning intended. This is the polysemy/false positives/low precision problem. Second, relevant items are missed when they are indexed not under the actual search terms, but rather under related terms. This is the synonymy/false negatives/low recall problem. With semantics-free approaches there is generally no way to improve both precision and recall at the same time. Increasing one is done at the expense of the other [Salton and McGill, 1983; van Rijsbergen, 1983]. For example, casting a wider net of search terms to improve recall of relevant items will also bring in an even greater proportion of irrelevant items, lowering precision.

There is a many-to-many mapping between word forms and word meanings. A single word form can have multiple meanings, and a single meaning can be expressed by multiple word forms. Both of these multiplicities cause problems for any approach to content search based on word forms. We believe that in order to do near-human level retrieval we must go beyond words and get at meanings. Text disambiguation during indexing should improve precision by combating polysemy [Krovetz and Croft, 1992]. We are looking into reducing the ambiguity of word forms during indexing by taking advantage of semantic networks. A number of these networks already exist and their implementation is fairly straightforward.

As part of a larger research project exploring the exploitation of explicit semantics for overcoming both the polysemy and synonymy problems, we have performed preliminary investigations of document indexing using a massive semantic network, WordNet. WordNet is a network of word meanings connected by a variety of lexical and semantic relations. Over 35,000 word senses are represented in the noun portion of WordNet alone. We have been working with WordNet in the SMART information retrieval environment. In the unconstrained text of the SMART environment, no index terms have been assigned [Buckley, 1985]. We have had to derive our own content description from the input text, given only part-of-speech tagging of the input.

Employing the notion of semantic distance between network nodes, we have run a series of experiments. Input text terms with multiple senses have been disambiguated by finding the combination of senses from a set of contiguous terms which minimizes total pairwise distance between senses. Results so far have been encouraging. Improvement in disambiguation compared with chance is clear and consistent, strongly suggesting that semantics-based indexing is worth pursuing further for transcending the polysemy problem. It is competitive with word-based approaches. A number of these have focused on only a few fixed terms whose senses were to be distinguished, rather than on unconstrained text [Lesk, 1986; Wilks *et al.*, 1989; Voorhees *et al.*, 1992].

In the following sections we will discuss the research environment, network-based disambiguation, the experiments performed and results obtained.

CIKM '93 - 11/93/D.C., USA

© 1993 ACM 0-89791-626-3/93/0011 ....$1.50

strike tap) which IS-A (impact bump thump blow); another maps to (strike work_stoppage) which IS-A (direct_action).

We work with the *Time Magazine* article collection, since it is the least specialized and technical, because WordNet is a general English lexicon.

With SMART, the words in the documents are converted to lower case and parsed into strings. They can be stemmed down to base forms; e.g., "stemmed" and "stems" both become "stem." Input words can also be labeled by part of speech, which is a feature that we took advantage of. Although the part-of-speech tagger employed was not infallible, it was accurate enough to give us a good working set of nouns to serve as input to semantic processing.

One aspect of this input editing process which is a source for limiting the effectiveness of our efforts is the filtering out of terms. SMART uses a list of "stopwords," words to be ignored as "contentless." For example, prepositions, conjunctions, and articles are considered extraneous. After stopwords have been removed, and non-nouns removed from what remains, very little of the original article is left. So, we are working with a sparse sample of the original text by the time we get to decide which sense of each noun is intended. Nouns found in WordNet are the final distillation that we begin to work with during disambiguation.

The following example illustrates the filtering process. It uses an excerpt from *Time* document 1, shown after successive filtering steps.

After conversion to lowercase (part-of-speech tagging is omitted for readability; the first four words are actually the title):

the allies after nassau in december 1960, the u.s . first proposed to help nato develop its own nuclear strike force . but europe made no attempt to devise a plan . last week, as they studied the nassau accord between president kennedy and prime minister macmillan, europeans saw emerging the first outlines of the nuclear nato that the u.s . wants and will support . it all sprang from the anglo-u.s . crisis over cancellation of the bug-ridden skybolt missile, and the u.s . offer to supply britain and france with the proved polaris (time, dec . 28).

After stopword removal:

allies . proposed nato develop nuclear strike force made attempt devise plan . week studied accord president kennedy prime minister macmillan emerging outlines nuclear nato . support sprang anglo crisis cancellation bug ridden skybolt missile offer supply britain france proved polaris time dec

```
synonymy  (has same meaning as; intranode)
hypernymy (is a)
hyponymy  (has instance)
holonymy  (is part of, is substance in,
           is member of; 3 relations)
meronymy  (has part, contains substance,
           has member; 3 relations)
antonymy  (is complement of; self-inverse)
```

Hypernymy and hyponymy are the strictly hierarchical links. The holonymy/meronymy relations can also be considered "vertical" relations. Vertical relations are asymmetrical and order items. Synonymy and antonymy are "horizontal," symmetrical, non-ordering relations (and of course are non-hierarchical).

3 Net-based Disambiguation

We have tried a variety of approaches to term disambiguation, all based on minimizing an objective function utilizing semantic distance between topics in WordNet. It is outside the scope of this paper to explain the distance determination logic. We will, however, describe the salient aspects of the network edge weighting scheme because this background is necessary for discussion of the experiments where the network weights were varied.

3.1 Edge weighting

Each edge consists of two inverse relations. Each relation type has a weight range between its own *min* and *max*. The point in the range for a particular arc depends on the number of arcs of the same type leaving the node. This is the *type-specific fanout* (TSF) factor. TSF reflects dilution of the *strength of connotation* between a source and target node as a function of the number of like relations that the source node has.[1] The two inverse weights for an edge are averaged. The average is divided by the depth of the edge within the overall "tree." This process is called *depth-relative scaling* and it is based on the observation that only-siblings deep in a tree are more closely related than only-siblings higher in the tree.

Definition 1

The edge between adjacent nodes A and B has distance or weight

[1] This factor takes into account the possible asymmetry between two nodes, where the strength of connotation in one direction differs from that in the other direction [Tversky, 1977].

nine internode relation types have preliminary weight ranges as follows: hypernymy, hyponymy, holonymy, and meronymy all have weights ranging from 1 to 2. Antonymy arcs all get the value 2.5 (there is no range).

3.2 Total distance minimization

We utilize semantic distance between network nodes, captured by the weights on the edges along the shortest path connecting the nodes, as a measure of relatedness between the topics represented by the nodes. The shorter the distance, the greater the relatedness. For disambiguation the hypothesis is that, given a set of terms occurring near each other in the text, each of which might have multiple meanings, by picking the senses that minimize distance we select the correct senses.

Overall distance minimization works as follows. For a given set of terms $T = \{t_1, t_2, ..., t_n\}$, each with possibly more than one candidate sense, each combination of n senses across the terms is tried, with one sense chosen at a time for each term. For example, given three terms t_1, t_2, t_3, with 2, 1, and 3 senses respectively, each of the $6 = 2 \cdot 1 \cdot 3$ combinations of senses is tried. For each combination of n senses, the pairwise distances between each pair of senses is found. The $\frac{n(n-1)}{2}$ pairwise distances are summed to arrive at an overall value, $H(T)$. The combination of senses which minimizes this sum is the "winning" combination.

Definition 2

For a set of neighboring terms $T = \{t_1, t_2, ..., t_n\}$, let $\mathcal{S}$ be the set of all combinations of term senses, which has cardinality $\prod_{i=1}^{n} |t_i|$, where $|t_i|$ is the number of senses of term i, and let $S \in \mathcal{S}$ be a particular combination of senses $\{s_1, s_2, ..., s_n\}$, where each s_j is a sense of t_j.

The winning combination is the $S \in \mathcal{S}$ which produces the minimal "energy"

$$H_{\min}(T) = \min_{S} \sum distance(x, y) \qquad \forall x, y \in S. \square^{2}$$

We call this technique *mutual constraint* among terms. There is a special case of mutual constraint where all terms except the one being disambiguated have had their senses determined and "frozen." Thus they have only one sense to work with now. When we are trying to disambiguate a term and work with previous frozen terms only, we speak of using a *frozen past* approach.

tially linear-time processing, since there are only as many "combinations" to try as there are senses of the single term being disambiguated.

Which term(s) gets its winning sense assigned varies depending on the type of window used. When working with a frozen past window of size n, only the $(n+1)$st term is assigned its sense. Each of the n window terms has already had its sense frozen. When working with a moving mutual constraint window, just the middle term is assigned its sense. Record is kept of the winning sense, but when that term plays a role other than "middle term," its senses are allowed to fully vary. This gives a middle term full benefit of both previous and subsequent context. All senses of surrounding terms are considered, not just their winning senses. For initial (as opposed to moving) mutual constraint windows, all of the terms in the window are assigned their senses at the same time.

4 Experiments

We have performed a number of disambiguation experiments with the *Time* collection. One series of experiments varied window size and type, and a second series varied network weighting schemes. Before discussing our experimental results, we need to cover the subject of measuring performance during disambiguation.

4.1 Performance evaluation

How do we measure success in disambiguation? We need to know what the "right" answer is for each term being disambiguated. This knowledge is provided by manual analysis and disambiguation of the terms. Because this is tedious and problematic work, we originally only hand-disambiguated the first five *Time* documents.

During that process it became evident that there are a number of situations that can arise when considering the input to the disambiguator. Seven situations can be distinguished:

1. There are multiple "good" senses — more than one sense of the input term is applicable in the context in which the term appears.

2. There is exactly one good sense.

3. There are no applicable senses. This has five variations:

   3a. The item is not actually a noun here (e.g. "prime" in "prime minister")

   3b. The item is a noun, but not the one the program sees (e.g. "cent" from "per cent")

[2] $distance(x,y)$ = $distance(y,x)$ = $\frac{distance(x\rightarrow y)+distance(y\rightarrow x)}{2}$ $distance(x,x) = 0$.

disambiguate, since any choice is a success. Only when at least one sense is good and at least one is bad can we consider picking a correct sense a success worth rewarding. Thus we focus on nontrivial terms — those which are true tests of disambiguation prowess.

One obvious way of evaluating success is to find the percentage of terms correctly disambiguated (out of the nontrivial terms). We will use this "hit-or-miss" measure as a secondary indicator. Since it does not reflect the difficulty present for individual terms, we have chosen to focus on another measure that takes this difficulty into account. This is the "hit score" — the ratio of "actual hit points" to "maximum hit points." Hit points are awarded as follows.

Definition 3

For each term let s be the number of senses and let g be the number of good senses (in context). The *hit points* for a hit are $s/g - 1$. Misses get zero points. □

The actual hit points for individual terms are summed, and this sum is divided by the sum of the maximum number of hit points possible, derived by treating all nontrivial terms as having been disambiguated correctly and their hit points awarded accordingly. Formally, hit score over n terms equals

$$\frac{\sum_{i=1}^{n} hitpoints_i \quad where \quad term_i \quad is \quad a \quad hit}{\sum_{i=1}^{n} hitpoints_i}.$$

Hit scores range from 0 to 1.

After manual disambiguation, the first five *Time* documents served as a standard against which to measure the performance of the semantic distance software. During manual disambiguation, the several situations that can arise for a term which were outlined above were taken into account when classifying the terms. The large majority of the terms had at least one good sense. Some basic quantities for the five documents are:

```
1175 terms remaining after stopword removal
 544 of those are nouns and in WordNet

122 type 1 terms (multiple good senses)
364 type 2 terms (one good sense)
 58 type 3 terms (no good senses) as follows:
 18 type 3a (not really a noun)
  4 type 3b (wrong noun)
  6 type 3c (unstemmed, taken as is)
  7 type 3d (proper noun)
 23 type 3e (sense not in WordNet)

486 possible hits (at least 1 good sense)
```

The standard deviation of the distribution of hit scores obtained from multiple runs of the "chance" software is approximately 0.04. In other words, taking a ± two standard deviation range, the "chance" software will give a hit score in the range 0.259 ± 0.08 with high probability. Thus if an alternative method scores well above 0.259 + 0.08 = 0.339, it is performing statistically significantly above the "chance" method.

These chance values were derived analytically and then verified empirically. For 20 empirical random sense selection runs the average hit score was between .25 and .26.

As "chance" provides a lower bound to compare our results against, human performance on the same tasks provides an upper bound. We had human subjects pick their estimate of the correct sense for each noun in WordNet for the first five *Time* documents. Two sets of printouts were distributed, each with the nouns in documents 1-5. Each noun's synset was given, along with its hypernym's synset and a gloss if available. The subjects were thus given roughly the same sparse information that the software was getting. Although the humans could bring to bear their world knowledge and linguistic knowledge, which should give them a large advantage, they were also handicapped by only receiving very local network data (node and parent only). In contrast, the software has the entire network at its disposal, albeit for its limited approach of looking at semantic distance. Also, the wording within synsets is quite terse and might not be highly suggestive of the actual sense intended. Thus, humans might find the information difficult to glean meaning from.

Averaging over the two tests, the average percent correct was .782 and the average hit score .706. Of course this sample is too small for statistical robustness. Nevertheless, it succeeds in giving us an idea of how people do under these same conditions.

For all of the experiments with the software, results are given for hit score unless otherwise stated. Generally hit score is more informative than simple percent correct.

4.2 Window variation

In the first series of experiments, window type and size were varied. First we tried frozen past windows of increasing size, from 1 to 100. These moving window results are given in Figures 1 and 2.

As one can see, success climbs to a point and then tapers off. This may be an effect of local discourse context size. As can be seen, the semantic distance approach produces results which are highly statistically significant. This is all the more significant, given the number of filters that the input text has gone through, and the amount of "noise"

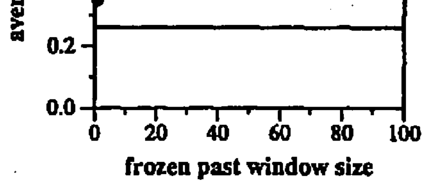

Figure 1: Comparison of hit scores for chance, semantic distance software, and human subjects for *Time* documents 1-5.

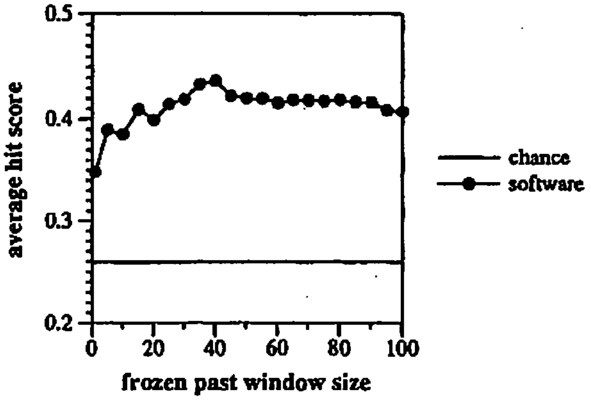

Figure 2: The same data as in Figure 1 but with the vertical scale restricted.

Figure 3: Pinning down the optimal moving frozen past window; no mutual constraint window.

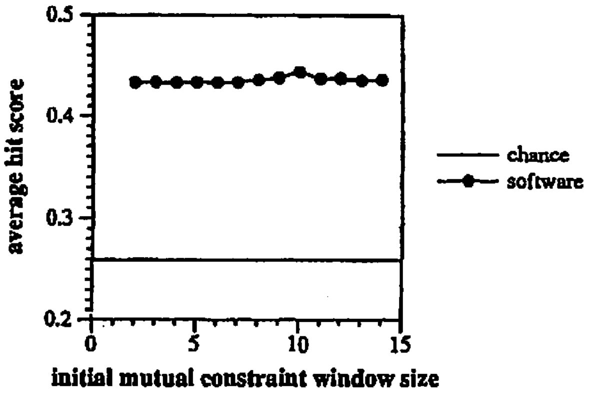

Figure 4: Initial mutual constraint window with frozen past window = 41.

in the remaining "signal." Also, since the semantic net resources used are relatively rudimentary compared to what they might be potentially, even greater success is possible.

The next experiments attempted to pin down the peak performance seen near window sizes of 35 and 40. The best result was with a frozen past window size of 41, .437525. See Figure 3.

Next, fixing the frozen past window size at 41, we tried augmenting this with an initial mutual constraint window. We were unable to proceed past an initial window size of 14 because the runs were taking exponentially longer. The best results were with an initial mutual constraint window of size 10, given the frozen past window of size 41 for all subsequent terms (henceforth "(10,41)"). The hit score was .446771. All terms within the initial mutual constraint window had their sense selections fixed simultaneously once the objective function had determined the winning combination of senses. See Figure 4.

We next tried a moving mutual constraint window. By the time we had made the window size 9, the runs were taking about three hours, so we stopped there. The results were tantalizing, as the hit scores were just getting above .4 at the point where we were forced to halt. See Figure 5.

Note that these runs take longer per window size than the ones where only the initial terms are processed using mutual constraint. The moving mutual constraint window

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase®
Smarter legal research.