Automatic Sense Disambiguation of the Near-Synonyms in a Dictionary Entry

Diana Zaiu Inkpen and Graeme Hirst

Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, M5S 3G4
{dianaz,gh}@cs.toronto.edu

Abstract. We present an automatic method to disambiguate the senses of the near-synonyms in the entries of a dictionary of synonyms. We combine different indicators that take advantage of the structure on the entries and of lexical knowledge in WordNet. We also present the results of human judges doing the disambiguation for 50 randomly selected entries. This small amount of annotated data is used to tune and evaluate our system.

1 Near-Synonyms

Near-synonyms are words with close senses. They are described in dictionaries such as *Webster's New Dictionary of Synonyms* (Gove 1984) and *Choose the Right Word* (Hayakawa 1994) (hereafter CTRW). An entry in these dictionaries presents a cluster of near-synonyms, explains the core meaning that they share, and makes explicit the differences between them. The differences include stylistic, attitudinal, and denotational nuances (see Edmonds 2000, Hirst 1995 for more details). An example of a fragment of an entry in CTRW[1] is presented in Fig. 1. CTRW contains 914 such entries.

We want to disambiguate the senses of the near-synonyms in each entry, as part of a bigger project which aims to automatically acquire knowledge of near-synonym differences from CTRW and other sources. A lexical knowledge-base of near-synonym differences is useful in an MT system to preserve not only the meaning of the sentences but also the nuances of meaning that words may carry and to avoid expressing unwanted nuances. Similarly, the lexical choice process in an NLG system can greatly benefit such information (Edmonds 2002). The first stage of the lexical acquisition process is presented by Inkpen and Hirst (2001).

In this paper, we present an algorithm for automatic sense disambiguation that takes advantage of the fact that the near-synonyms can help disambiguate each other, and the text of the entry is a rich context for disambiguation. We also present the agreement among judges in the task of annotating a small amount of data, which we use to both tune and evaluate our system.

2 Sense Disambiguation

Sense disambiguation means to select one or more senses in which a word is being used in a particular context. The task of disambiguating the meaning of near-synonyms is

[1] We are grateful to HarperCollins Publishers, Inc. for permission to use CTRW in our project.

Cluster: **acumen, acuity, insight, perception**
These nouns all refer to a highly developed mental ability to see or understand what is not obvious. **Acumen** has to do with keenness of intellect and implies an uncommon quickness and discrimination of mind. It requires acumen to solve an intricate problem in human relationships, or to emerge unscathed from a venture into penny stocks. . . . .
Acuity means sharpness or keenness, and is applied exclusively to perception: visual acuity; The intelligence test was used as a basis for judging the applicant's mental acuity. *See* KEEN, SENSATION, VISION, WISDOM. *Antonyms*: bluntness, dullness, obtuseness, stupidity.

Fig. 1. Part of the text of an entry in *Choose the Right Word* by S.I. Hayakawa. Copyright ©1987. Reprinted by arrangement with HarperCollins Publishers, Inc.

easier than the general task of word sense disambiguation (WSD). But it is not a simple task. As we show in section 4, disambiguating the meaning of the near-synonyms is not easy even for humans.

In our task the context is richer than in the general case. We can use the full text of each entry (including the cross-references). For each entry in CTRW, we consider all senses of each near-synonym. We chose to use WordNet1.7 sense inventory in order to integrate our word sense disambiguation program with other components in our project that use WordNet. The average polysemy for CTRW is 3.18 (for 5,419 near-synonyms there are 17,267 WordNet senses).

For each sense we need to decide whether it is relevant for the entry or not. For example, in Fig. 1, *acumen* has two WordNet senses: `acumen#n#1` glossed as "a tapering point", and `acumen#n#2` glossed as "shrewdness shown by keen insight". The decision we want to make is that the second one is relevant for the entry, and the first one is not. More then one sense of a near-synonym can be relevant for an entry, so we view the problem as one of binary decisions: for each sense, decide whether it is relevant for the context or not. To disambiguate each sense, we compute the indicators described below. Then we combine them to decide if the sense is relevant.

2.1 Intersection of text and gloss

Our main indicator is the size of the intersection of the text of the entry with the WordNet gloss of the sense, both regarded as bags of words. This is a Lesk-style approach (Lesk 1986). When we intersect the text with the gloss we ignore stopwords and the word to be disambiguated. (We experimented with stemming the words, but it does not improve the results.) The other near-synonyms occur in the text of the entry; if they happen to occur in the gloss, this is a good indication that the sense is relevant.

Sometimes the intersection contains only very common words that do not reflect a real overlapping of meaning. In order to avoid such cases, we weight each word in the intersection by its *tf·idf* score. The weight for the word i in the entry j is $tf{\cdot}idf_{i,j} = n_{i,j} \log \frac{n_i}{N}$, where $n_{i,j}$ is the number of occurrences of the the word i in the entry j, n_i is the number of entries that contain the word i, and N is the total number of entries. While we could have imposed a general threshold for the intersection (if the score is lower than the threshold, the sense is not relevant), we preferred to train a decision tree to choose a series of thresholds to better fit the data (see section 2.6).

We also intersected the text of the entry with the glosses of related words. The hyponym/hypernym glosses can be expected to work well because some of the near-synonyms in CTRW are in a hypernymy/hyponymy relation with each other. We also exploited other relations available in WordNet, such as meronymy, holonymy, pertainymy for adjectives, cause and entailment for verbs, etc.

2.2 Other words in synsets being near-synonyms

Our next indicator is the other words in each synset. They reliably indicate a sense being relevant for the entry because the near-synonyms in the entry help disambiguate each other. For example, if the cluster is: *afraid, aghast, alarmed, anxious, apprehensive, fearful, frightened, scared*, when examining the senses of *anxious*, the sense corresponding to the synset `anxious#a#1`, `apprehensive#a#2` is relevant because the other word in the synset is *apprehensive*, which is one of the near-synonyms.

We also used the words in synsets of related words. If any of the words in the synsets of the words related to the sense under consideration happens to be a near-synonym in the same cluster, the sense can be judged as relevant. Again, by related words, we mean words connected by a direct WordNet relation.

2.3 Antonyms

The antonyms in the entry are intersected with the antonyms of the current near-synonym. Fig. 1 shows an example of antonyms for a cluster. If two words share an antonym, they are likely to be synonyms. By extension, if the sense we examine has antonyms that intersect the antonyms of the cluster, then the sense is relevant for the cluster.

2.4 Systematic polysemy

We tested the hypothesis that if a word is polysemous in a systematic way, all its senses are included in a dictionary entry. We did this for nouns only, using CoreLex (Buitelaar 1998), a database of systematic polysemous classes covering around 40,000 nouns from WordNet 1.5. The 126 semantic types are derived by a careful analysis of sense distributions. We experimented with both the original CoreLex, and with a new version of CoreLex we built for WordNet1.7 using the same set of semantic types. This indicator selects as relevant all the senses of a noun that is in CoreLex (and therefore is systematically polysemous).

2.5 Context vectors

Sometimes the intersection of text and gloss is empty when there is similarity but different words are used. By considering second-order co-occurrences (words that occur with the words of the text or of the gloss) we increase the chance of detecting similarity (Schütze 1998).

We collected frequencies from the 100 million-word British National Corpus (BNC) (http://www.hcu.ox.ac.uk/BNC/). We chose the 2,000 most frequent words as dimensions, and the 20,000 most frequent words as features. By counting how many times

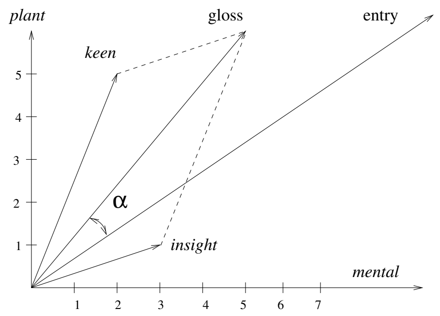

Fig. 2. Context vectors in a 2D space

each feature word co-occurs with a dimension word, we can represent them in the vector space of the dimensions. Then, the vectors of all feature words in an entry (except the near-synonym to be disambiguated) are summed to compute the context vector for the entry. The vectors of all words in a gloss are summed to get the context vector for the gloss. The cosine between the two vectors measures how close the two vectors are. The context vector for the entry will be the sum of many vectors, and it may be a longer vector than the context vector for the gloss, but this does not matter because we measure only the angle between the two vectors. Fig. 2 presents a simplified example of context vectors for the second sense of *acumen*. For simplicity, only two dimensions are represented: *plant* and *mental*. Also, from the four content words in the gloss, three happen to be feature words, and only *keen* and *insight* are presented in the figure. The context vector of the gloss is sum of these two (or more) vectors. In a similar manner the context vector for the entry is obtained, and the cosine of the angle α between the two context vectors is used as an indicator for the relevance of the sense. Here, the cosine is 0.909, while the cosine between the context vector for the entry and the context vector for the first sense of *acumen* is only 0.839.

2.6 Using a decision tree to combine indicators

We use decision tree learning to determine the best combination of indicators. We use C4.5 (http://www.cse.unsw.edu.au/∼quinlan/) on our 904 data points, using seven attributes: intersection text and gloss (numerical value), intersection of text and gloss of related words (numerical value), words in synset (0 or 1), words in synsets of related words (0 or 1), antonyms (0 or 1), membership in CoreLex (0 or 1), and the cosine between context vectors (numerical value). The classification is binary: Y/N, meaning relevant or not relevant for the entry. See Fig. 3 for a simplified decision tree.

We experimented with manual combinations, but we decided it is better to automatically derive a decision tree, because this learning mechanism has the ability to decide which indicators have more influence on the classification, and it can completely ignore indicators with low influence. We could have split the annotated data into training and test data, but we chose to do 10-fold cross-validation, as a better method to estimate

```
intersection_text_gloss > 4.41774 : Y (406.0/59.4)
intersection_text_gloss <= 4.41774 :
| intersection_text_gloss_related_words > 23.7239 : Y (28.0/1.4)
| intersection_text_gloss_related_words <= 23.7239 :
| | words_in_related_synsets = 0:
| | | words_in_synset = 0:
| | | | intersection_text_gloss_related_words <= 4.61842 : N (367.0/62.5)
| | | | intersection_text_gloss_related_words > 4.61842 :
| | | | | intersection_text_gloss_related_words <= 4.94367 : Y (4.0/1.2)
| | | | | intersection_text_gloss_related_words > 4.94367 : N (42.0/14.6)
| | | words_in_synset = 1:
| | | | intersection_text_gloss <= 1.19887 : N (16.0/8.9)
| | | | intersection_text_gloss > 1.19887 : Y (3.0/1.1)
| | words_in_related_synsets = 1:
| | | intersection_text_gloss <= 0 : Y (24.0/4.9)
| | | intersection_text_gloss > 0 :
| | | | corelex = 0:
| | | | | cosine <= 0.856407 : Y (3.0/2.1)
| | | | | cosine > 0.856407 : N (5.0/1.2)
| | | | corelex = 1:
| | | | | intersection_text_gloss <= 3.44834 : Y (3.0/1.1)
| | | | | intersection_text_gloss > 3.44834 : N (3.0/2.1)
```

Fig. 3. Simplified decision tree

the error rate. Another advantage of using the decision tree is that it determines the thresholds for weighted intersection and for cosine.

3 Results and Evaluation

Table 1 presents the results of using each indicator alone and in combinations with other indicators. We compare the results of our method with a standard solution consisting of the annotated data whose creation we describe in section 4 below. For the most of the indicators, we use the standard solution to quantify their potential.

For the indicators using *tf·idf* and for the cosine between context vectors we use a decision tree to avoid manually choosing a threshold; therefore the figures in the table are the results of the cross-validation. By manually combining indicators, the best accuracy we obtained was 79.3% for the attributes: intersection text and gloss (with a fixed threshold), words in synsets, and antonyms.

We found the best combination of indicators by training a decision tree as described in section 2.6. We achieve an accuracy of 83%, computed by 10-fold cross-validation. The indicators that contribute the most to improving the accuracy are the ones in the upper-part of the decision tree (see Fig. 3): the intersection of the text with the gloss, the intersection of the text with the glosses of the related words, the words in the synset, and the words in the synsets of the related words. The ones in the lower part (CoreLex and the cosine between context vectors) have less influence on the results. Their contribution is likely to be included in the contribution of the other indicators.

4 Agreement among Judges

The goal of using human judges in our work was twofold: to get a measure of how difficult is the task for humans, and to build a standard solution for use in evaluation.

Method	Accuracy
Baseline (select all senses)	53.5%
Antonyms	47.0%
Cosine (decision tree)	52.7%
CoreLex	54.2%
Words in synsets of hypernyms and hyponyms	56.4%
Intersection text & gloss of hypernyms and hyponyms (*tf·idf*)	61.0%
Words in synsets of related words	61.3%
Words in synset	67.1%
Intersection text & gloss of related words (*tf·idf*) (decision tree)	70.6%
Intersection text & gloss (no *tf·idf*)	76.8%
Intersection text & gloss (*tf·idf*) (decision tree)	77.6%
Best combination (no decision tree)	79.3%
Best combination (decision tree)	**83.0%**

Table 1. Accuracy for different combinations of indicators

We had $k = 6$ judges (native or near-native speakers of English) doing the same job as the WSD program. We randomly selected 50 of the 914 clusters, containing 282 near-synonyms with 904 senses in total. The judges were presented with the text of the entry for each cluster, including antonyms and cross-references. For each near-synonym, all the WordNet senses (with their glosses and all the words in the synset) were listed, and the judges had to decide whether the sense is relevant for the cluster or not. The judges had no information about hypernyms, hyponyms, or antonyms.

There were 904 decisions the judges had to make. If we consider the decisions as votes, for 584 decisions, the judges voted 6–0 (or 0–6), for 156 decisions 5–1, and for 108 decisions 4–2. There were 56 ties (3–3).

To measure the agreement among the k judges, we used the well-known kappa statistics (Siegel and Castellan 1988), (Carletta 1996). The agreement among our the judges was 85%. The probability of agreement by chance is 50.2%. Therefore the kappa coefficient, which takes into account the possibility of agreement by chance, is $K = 0.699$. The figures of agreement between pairs of two judges vary from 90% ($K = 0.80$) to 78.8% ($K = 0.57$). If we leave out one of the judges, who expressed a particular bias, we get a higher agreement of 86.8% ($K = 0.73$).

We had the judges meet to discuss the ties. The discussion had a very small influence on the agreement figures (because the number of cases discussed was small), but it helped clarify the sources of disagreement. Senses which are "instances" or "proper names" (e.g. the sense "*the United States*" for the near-synonym *union*) were rejected by some judges as too specific, even if they were mentioned in the text of the entry. There was disagreement about intransitive senses of some transitive verbs (or the other way around). Another problem was posed by mentions of extended senses (or literal, or figurative senses) in the text. It is not always clear whether the extended sense is mentioned by the lexicographer who designed the entry, because the extended sense is very close and should be included in the meaning of the cluster, or whether it is mentioned so that the reader will distinguish it. In such cases, some judges decided to include more often than exclude, while the other judges excluded the senses when they thought appropriate.

In the standard solution, we decided to correct few of the 56 cases of ties, to correct the apparent bias of some judges. We decided to include senses that are too specific or instances, but to exclude verbs with wrong transitivity. We produced two solutions: a more inclusive one (when a sense is mentioned in the entry, it was included) and a more exclusive solution (when a sense is mentioned, it was included only if the judges included it). The more inclusive solution was used in our experiments, but the results should change very little if we use the more exclusive one, because they differ only in 16 points out of 904.

5 Comparison with Related Work

Senseval (http://www.itri.brighton.ac.uk/events/senseval/) had the goal of evaluating WSD systems. We will refer here only to the experiments for the English language. Senseval1 used the sense inventory from HECTOR project, while Senseval2 used WordNet1.7 senses. For this reason we can compare our results with the ones from Senseval2. Obviously the words and texts are different because of the nature of our task, so the comparison can be done only at a very general level. Senseval2 had two tasks: all content words and selected words (the last one seems closer to our task). The precision and recall reported by the participating systems were all below 70%, and fancy supervised or unsupervised algorithms could beat a Lesk-baseline by only 2%.

Our WSD program performs at least 13% better, because our task is relatively easier than the general WSD task. We report accuracy figures, but this is equivalent to reporting precision and recall, since we disambiguate all the near-synonyms. Our value for inter-annotator agreement (85%) is comparable to that of Senseval2 (85.5% for the English lexical sample task, according to the Senseval2 webpage).

Stevenson and Wilks (2001), among others, combine classifiers (knowledge-sources) by a using a weighted scheme. Our methods of combining classifiers is better, but requires a small amount of annotated data.

An adapted Lesk-style algorithm for WSD using WordNet is presented by Pedersen and Banerjee (2002). They intersected glosses of all words in the context of a target word. The intersection is done pairwise, also considering intersections between glosses of a word and words related to the second word (by hypernymy, hyponymy, meronymy, etc.). They achieve an accuracy of 32%. Unlike Pedersen and Banerjee, we focus only on the target word (we do not use glosses of words in context), when we use the gloss of a near-synonym we include examples in the gloss, and we achieve high accuracy.

Schütze (1998) uses context vectors to cluster together all the contexts in which a word is used in the same sense. In this way it is possible to distinguish among word senses without using a sense inventory from a lexical resource. We use the context vectors as a measure of the semantic relatedness between the text of an entry and the gloss of a synset.

6 Conclusion and Future Directions

We have presented a method to disambiguate senses of the near-synonyms in dictionary entries. We also presented the inter-annotator agreement for the human task and analyzed the sources of disagreements. We built a standard solution and used it to tune and evaluate our automatic program.

We plan to reuse our WSD program in other components of our system. For example, nouns that describe situations have associated semantic roles; we can extract them by finding verb senses with the same meaning as the nouns. Moreover, we can use a similar WSD algorithm for disambiguating senses of peripheral concepts (nuances) expressed by near-synonyms.

Acknowledgments

We thank Eric Joanis, Jane Morris, Alex Budanitsky, and ZuZu Gadallah for participating in the judging task. Our work is financially supported by the Natural Sciences and Engineering Research Council of Canada and the University of Toronto.

References

Buitelaar, Paul: An ontology of systematic polysemous classes. In *Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS'98)*, Trento (1998)

Carletta, Jean: Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2) (1996) 249–254

Edmonds, Philip and Hirst, Graeme: Reconciling fine-grained lexical knowledge and coarse-grained ontologies in the representation of near-synonyms. In *Proceedings of the Workshop on Semantic Approximation, Granularity, and Vagueness*, Breckenridge, Colorado (2000)

Edmonds, Philip and Hirst, Graeme: Near-synonymy and lexical choice. *Computational Linguistics*, 28(2) (2002) (to appear).

Gove, P.B. (ed.): *Webster's New Dictionary of Synonyms*. G.&C. Merriam Co. (1984)

Hayakawa, S.I.: *Choose the Right Word*. HarperCollins Publishers (1994)

Hirst, Graeme: Near-synonymy and the structure of lexical knowledge. In *Working notes, AAAI Symposium on Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, Stanford University (1995) 51–56

Inkpen, Diana and Hirst, Graeme: Building a lexical knowledge-base of near-synonym differences. In *Proceedings of the Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh (2001)

Lesk, Michael: Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of SIGDOC Conference*, Toronto (1986)

Pedersen, Ted and Banerjee, Satanjeev: An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2002)*, Mexico City, Mexico (2002)

Schütze, Hinrich: Automatic word sense discrimination. *Computational Linguistics*, 24(1) (1998) 97–123

Siegel, Sidney and Castellan, John N.: *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, Inc. (1988)

Stevenson, Mark and Wilks, Yorick: The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics*, 27(3) (2001) 321–350