Combination of contextual features for word sense disambiguation: LIU-WSD

Magnus MERKEL & Mikael ANDERSSON

Dept. of Computer and Information Science,

Linköping University

S581 83 Linköping, Sweden,

magme@ida.liu.se, miand@ida.liu.se

Abstract

This paper describes a system for word sense disambiguation that participated in the Swedish Lexical Sample task of SENSEVAL-2. The system LIU-WSD is based on letting different contextual features cast votes on preferred senses according to a ranking scheme.

Introduction

The addition of new languages to the SENSEVAL-2 workshop, among these languages also Swedish, presented an opportunity to learn more about WSD applied to Swedish by participation in the event. Previously, we had had no experience of building word sense disambiguation software, but the Swedish Lexical Sample task seemed like a suitable occasion for trying another field of NLP (in recent years our focus has been on word alignment and parallel corpora).

Due to time constraints our initial plans of implementing some kind of version of decision lists (Yarowsky, 2000; Pedersen 2001) were abandoned in the end and we decided to go for a slightly simpler approach based on a general algorithm and voting strategies for contextual features on different levels. The contextual features that were being considered were unigrams and bigrams, both in fixed and variable positions, together with possibilities to include parts-of-speech, lemmas and graph words (inflected words).

1 Data and pre-processing

The data and resources used in the LIU-WSD system were apart the following:

- Sample and training data, provided by the task organisers – the sample data were a great help in order to understand the format of the provided material.
- Part of the lexicon data provided for the Swedish lexical sample task. Here only the information on the number of senses and division of main and sub senses was used. Information contained in examples, definitions and for valency was left out.

As the system was a first attempt to word sense disambiguation for us, we set up a small corpus containing for five lexical items with around 60 contexts for each lexeme. This was used to test various approaches and algorithms as well as making sure of conversions to and from the format used in the task.

As we wanted to use morphosyntactic information and lemmas in the system, the Swedish Constraint Grammar package, SWECG-2 from Conexor was used (Karlsson et al., 1994). The training corpus was fed into the tagger, SWECG-2, which returned an XML file where the text was POS tagged and lemmatised. Below is a sample of how the sentence *Men påståendet är litet missvisande.* came back in XML format.

```
<instance id="barn.19">
<answer instance="barn.19"/>
<context>
<w id="w1" base="men" pos="CC">men</w>
<w id="w2" base="påstående"
    pos="N">påståendet</w>
<w id="w3" base="vara" pos="PRES">är</w>
<w id="w4" base="litet & litet" pos="ADV &
    A">litet</w>
<w id="w5" base="missvisa"
    pos="NDE">missvisande</w>
  <w id="w6" base="."
    pos="interpunction">.</w>...
```

2 Training

The actual training was a matter of building tables of information from the training corpus. The task was first to decide on a number of contextual features to be observed, then set

```
 SENSES    frq| 1   1.a 1.b 1.c 1.d 1.e 1.f 1.g  2  2.a 2.b 2.c 2.d  3   4
 Distrib.  152|  30  12   0   7  15   4   1  14   6   7   0  32   3   2  19
 Rel.freq.    | .19 .07  -  .04 .09 .02  -  .09 .03 .04  -  .21 .01 .01 .12
 Stand.dev.   | .03 .02  -  .01 .02 .01  -  .02 .01 .01  -  .03 .01  -  .02
 ------------------------------------------------------------------------
 egen        4|  4   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 Rel.freq.    | 1.0  -   -   -   -   -   -   -   -   -   -   -   -   -   -
 Ratio        |  5   -   -   -   -   -   -   -   -   -   -   -   -   -   -
 T-score      |24.8  -   -   -   -   -   -   -  '-   -   -   -   -   -   -
```

Table 1. Extract from training data for the lexeme *kraft*, and the contextual feature [-1, lemma] when *egen* is observed in position -1. The upper part of the table shows the sense distribution in the training corpus and the lower part give data on *egen* when seen immediately in front of *kraft*.

thresholds and store the information as a database where each lexeme was represented with a number of relevant contextual features observed in the training data.

Basically, a table contains general information on the sense distribution in the training corpus, including frequencies, relative frequencies, standard deviation and a statistical measure, the Student T-measure. The idea behind the statistical measure is to use a test that can tell whether a certain observation of a contextual feature in relation to the choice of a word sense is statistically significant or not. We test whether the existence of a certain contextual feature F changes the distribution of senses for a certain sense S. There are many tests that can be used. We chose to use the Student T-test. The

$$p = \frac{s}{T}$$

probability p for a sense S is estimated as:
where s is the total number of contexts holding word W with sense S, and T is the total number of contexts where the word W occurs.

To arrive at a t-score we set N as the number of contexts for W where the feature F is holds, and n as the number of contexts among N that contain the sense S. We then calculate p':

$$p' = \frac{n}{N}$$

To be interesting p' must be greater than p. We can now test whether the distribution for F is the same as if when F is not observed, which is the actual t-test. We then test $H0$ (p=p') vs. $H1$ (p<p') and calculate t as

$$t = \frac{p'-p}{dev}$$

where dev is the standard deviation for p, testing on the 95% confidence level. An example how the t-score looks for a particular feature is illustrated in Table 1.

2.1 Feature patterns

Twelve general feature patterns were extracted from the training corpus. The patterns are defined by choosing options on three different levels: (i) unigram or bigram, (II) lemma (base form), graph word (inflected form) or parts-of-speech category (POS), and (iii) fixed position or position within an interval.[1]

1. Unigram, lemma, position -1
2. Unigram, lemma, position +1
3. Unigram, lemma, position -5 to -2
4. Unigram, lemma, position +2 to +5
5. Unigram, lemma, all positions
6. Unigram, POS, position -2 to -1
7. Unigram, POS, position +1 to +2
8. Bigram, lemma, position -2 to -1
9. Bigram, lemma, position +1 to +2
10. Bigram, lemma, all positions
11. Bigram, POS, position -2 to -1
12. Bigram, POS, position +1 to +2

Patterns 1-7 all concern unigrams, while the rest operate on bigrams. Pattern 11, for example, will extract all POS bigrams in position -2 to -1, i.e., the POS tags for the two words that immediately precede the head word (which is assumed to be in position 0).

For each lexeme, a table was built like the one in Table 1 for each of the twelve general feature

---

[1] Feature 3, 4, 5, 6, 7 and 10 are all be bags of word features. The others indicate fixed positions.

patterns. The frequency threshold used was 3 and function words were ignored except in the bigram patterns.

3 Procedure and algorithm

When a test instance is to be disambiguated, a pre-processor first matches the test context with the training data for the lexeme in question. This involves identifying exactly those contextual features from the test data that are applicable to the test instance. The filtered set of tables is then used as input to the main disambiguation algorithm.

The algorithm is based on a voting strategy combined with some heuristics. The voting strategy entails that all classes of feature patterns (at the most 12 classes) will cast votes for a particular sense. The class vote is determined by which sense is ranked highest within that class. The winner for each class is determined by the number of sense choices for the included features of that class. For example, for the instance 9 in the test corpus of the lexeme *barn* (Eng. *child*), the voting would result in choosing sense *barn_1_1* as it was ranked as number one in three classes, see below.

```
barn.9:
VOTES for senses:
Sense
   1_1:   Rank1: 3
   1_1.a: Rank3: 1
   1_1.b: Rank1: 1, Rank3: 1
   1_2:   Rank1: 1, Rank2: 1
   1_2.a: Rank3: 1
sense selected: barn_1_1
```

During the training phase it was discovered that features that contained a relative frequency of 1.0 (i.e. all observations of the feature only occurred with a single sense) could be considered as a relatively "sure sense", we included this strategy in the algorithm.

The basic outline of the algorithm is as follows:

1. If there are no applicable data for the instance (i.e. no tables), pick the most common sense in the training corpus as the sense for this instance.
2. Otherwise, if there is a "sure sense", i.e. a contextual feature is found with the relative frequency 1.0, this sense is selected, if there is no conflict between several "sure senses".
3. Check the t-scores for all features in every class, and rank each sense as Rank1, Rank2, Rank3, etc. Each feature class will then be ranked and will cast votes accordingly. If there is a sense winner (i.e. most number of first places in the feature classes), this sense is selected.
4. If there are several senses that are tied, check how many votes they have for second places, and the best one of them is chosen as the sense.
5. If there's still a tie when considering second places, start clustering senses together into a main sense, for example, 1_1.a, 1_1_b, and 1_1_c are all considered as sense 1.1. Only those senses that were tied for first place are considered, but if these senses all share a single main sense, then that sense is chosen.
6. If there be several main senses, go back to original (sub-)senses, and select the sense with highest frequency in the training corpora.
7. If all of the above fails, simply resort to taking the most common sense in the training sample.

4 Results

The official results for the LIU-WSD system in the Swedish Lexical Sample task were the following:

Fine-grained scoring: 56.5 per cent precision, 56.5 per cent recall.
Mid-grained scoring: 61.6 per cent precision, 61.6 per cent recall.

The coarse-grained scoring is not relevant as an evaluation criterion due to the fact that the tested lexemes were categorised as belonging to the same main sense, which means that all results received precision scores of 100 per cent.

5 Evaluation

As the results were slightly worse than we had hoped for, it is interesting to point out further details from the scores and on the strategies that were actually used by the system. Table 2 illustrates what part of the algorithm that was used for selecting a particular sense and how well each strategy worked. It is notable that

Table 2. Overview of sense selecting criteria for the LIU-WSD system in SENSEVAL-2

	TOTAL	CORRECT	ERROR	PRECISION
Sure senses:	875	619	256	70%
Voted Rank1	393	194	199	49%
Tied Rank2	115	28	87	24%
Common main sense	27	7	20	25%
Most frequent main sense:	38	7	31	18%
Sub sense	77	17	60	22%
Most frequent sense	0	0	0	
	1525	872	653	

more than 50 per cent of the selections were made by the heuristics to select a "sure sense" based on the relative frequency. This is also the most successful in terms of precision of all the strategies. The voting strategies performed far worse, 49% for selection based on senses that were ranked first, and only 24% when a tie for first place was found and the second positions were considered. The strategies to select a shared common main sense, a most frequent main sense or most frequent sub sense when the other criteria failed were clearly not very successful, as indicated by precision rates varying from 18 to 25 per cent.

It is also worth pointing out that there were always some significant features for each instance of the test corpus, which meant that step 1 of the algorithm never triggered. The same is true for the last step. If we break down the results into different parts-of-speech, we can see the following:

Table 3. Results for nouns, verbs and adjectives

NOUNS	
Precision:	74%
Better than baseline (MFS):	20/20
Average no. of senses:	8.05
VERBS	
Precision:	40.4%
Better than baseline (MFS):	12/15
Average no. of senses:	14.26
ADJECTIVES	
Precision:	40%
Better than baseline (MFS):	4/5
Average no. of senses:	14.4

As has been noted elsewhere, nouns are easier than verbs and adjectives when it comes to word sense disambiguation (cf. Yarowsky 2000). This is clearly the case here, and a contributing factor to this is that the number of senses for nouns is significantly smaller than for the other word classes. However, the system does in general perform better than the standard baseline (Most Frequent Sense of the training corpus). For nouns, all 20 test instances are better than the baseline. There is however work to be done to improve the performance for verbs and adjectives.

6   Discussion

Clearly this system can be improved further, especially when it comes to how the voting system should be set up. As the feature classes sometimes are overlapping, some features will contribute several times to the votes (but from different classes), therefore some kind of inductive Machine Learning algorithm to infer which combination of features is the best should be tested. Another possible improvement would be to include information from the examples in the lexicon and also to include the inflected form of the word that is to be disambiguated in the process.

References

Karlsson, F,. A. Voutilainen, J. Heikkilä & A. Antilla (1994). *A Language-Independent System for Parsing Unrestricted Text.* Mouton de Gruyter, Berlin and New York.

Pedersen T. (2001). A Decision Tree of Bigrams is an Accurate Predictor of Word Sense. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*, Pittsburgh, PA.

Yarowsky D. (2000). Hierarchical Decision Lists for Word Sense Disambiguation. *Computers and the Humanities*, 34(2):179-186, 2000