7 *Word Sense Disambiguation*

THIS CHAPTER gives an overview of work on word sense disambiguation within Statistical NLP. It introduces a few of the most important word sense disambiguation algorithms, and describes their resource requirements and performance.

What is the idea of word sense disambiguation? The problem to be solved is that many words have several meanings or *senses*. For such words given out of context, there is thus *ambiguity* about how they are to be interpreted. As a first example of ambiguity, consider the word *bank* and two of the senses that can be found in Webster's New Collegiate Dictionary (Woolf 1973):

SENSES

AMBIGUITY

- the rising ground bordering a lake, river, or sea ...
- an establishment for the custody, loan exchange, or issue of money, for the extension of credit, and for facilitating the transmission of funds

DISAMBIGUATION

The task of *disambiguation* is to determine which of the senses of an ambiguous word is invoked in a particular use of the word. This is done by looking at the context of the word's use.

This is how the problem has normally been construed in the word sense disambiguation literature. A word is assumed to have a finite number of discrete senses, often given by a dictionary, thesaurus, or other reference source, and the task of the program is to make a forced choice between these senses for the meaning of each usage of an ambiguous word, based on the context of use. However, it is important to realize at the outset that there are a number of reasons to be quite unhappy with such a statement of the task. The word *bank* is perhaps the most famous example of an ambiguous word, but it is really quite atypical. A more typical situation is that a word has various somewhat related senses, and it is unclear whether to and where to draw lines between them. For example, consider the word *title*. Some senses that we found in a dictionary were:

- Name/heading of a book, statute, work of art or music, etc.
- Material at the start of a film
- The right of legal ownership (of land)
- The document that is evidence of this right
- An appellation of respect attached to a person's name
- SYNECDOCHE A written work [by *synecdoche*, i.e., putting a part for the whole]

One approach is simply to define the senses of a word as the meanings given to it in a particular dictionary. However, this is unsatisfactory from a scientific viewpoint because dictionaries often differ greatly in the number and kind of senses they list, not only because comprehensive dictionaries can be more complete, but fundamentally in the way word uses are gathered into senses. And often these groupings seem quite arbitrary. For example, the above list of senses distinguishes as two senses a right of legal title to property and a document that shows that right. However, this pattern of sense extension between a concept and something that shows the concept is pervasive and could have been, but was not, distinguished for other uses. For example the same ambiguity exists when talking about the title of a painting. For instance, one might remark in a gallery:

(7.1) This work doesn't have a title.

That sentence could mean either that the work was not given a title by the author, or simply that the little placard giving the title, which usually appears by paintings in a gallery, is missing. It is also somewhat unclear why books, statutes and works of art or music are grouped together while films are separated out. The second definition could be seen as a special case of the first definition. It is quite common in many dictionaries for senses to be listed that are really special cases of another sense, if this sense is frequently and distinctively used in texts. These difficulties suggest that, for most words, the usages and hence the sense definitions are not to be thought of as like five kinds of cheese, among which one must choose, but more like a casserole which has some pieces of clearly distinct identifiable content, but a lot of stuff of uncertain and mixed origin in between.

Notwithstanding these philosophical objections, the problem of disambiguation is of clear importance in many applications of natural language processing. A system for automatic translation from English to German needs to translate *bank* as *Ufer* for the first sense given above ('ground bordering a lake or river'), and as *Bank* for the second sense ('financial institution'). An information retrieval system answering a query about 'financial banks' should return only documents that use *bank* in the second sense. Whenever a system's actions depend on the meaning of the text being processed, disambiguation is beneficial or even necessary.

There is another kind of ambiguity, where a word can be used as different parts of speech. For example, *butter* may be used as a noun, or as a verb, as in *You should butter your toast.* Determining the usage of a word in terms of part of speech is referred to as *tagging*, and is discussed in chapter 10. How do these two notions relate? Using a word as a verb instead of as a noun is clearly a different usage, with a different meaning involved, and so this could be viewed as a word sense disambiguation problem. Conversely, differentiating word senses could be viewed as a tagging problem, but using semantic tags rather than part of speech tags. In practice, the two topics have been distinguished, partly because of differences between the nature of the problem, and partly because of the methods that have been used to approach them. In general, nearby structural cues are most useful for determining part of speech (e.g., is the preceding word a determiner?), but are almost useless for determining semantic sense within a part of speech. Conversely, quite distant content words are often very effective for determining a semantic sense, but are of little use for determining part of speech. Consequently, most part of speech tagging models simply use local context, while word sense disambiguation methods often try to use content words in a broader context.

TAGGING

The nature of ambiguity and disambiguation changes quite a bit depending on what material is available for training a word sense disambiguation system. After an initial section about methodology, this chapter has three main sections dealing with different types of training material. Section 7.2 describes *supervised disambiguation*, disambiguation based on a labeled training set. Section 7.3 describes *dictionary-based disambiguation*, disambiguation that is based on lexical resources such as dictionaries and thesauri. Section 7.4 deals with *unsupervised disam-*

biguation, the case in which only unlabeled text corpora are available for training. We conclude with an in-depth discussion of the notion of sense and pointers to further reading.

7.1 Methodological Preliminaries

Several important methodological issues come up in the context of word sense disambiguation. They are of general relevance to NLP, but have received special attention in this context. These are: supervised vs. unsupervised learning; the use of artificial evaluation data, known in the word sense disambiguation context as *pseudowords*; and the development of upper and lower bounds for the performance of algorithms, so that their success can be meaningfully interpreted.

7.1.1 Supervised and unsupervised learning

SUPERVISED LEARNING
UNSUPERVISED LEARNING
CLUSTERING
CLASSIFICATION

A lot of algorithms are classified as to whether they involve supervised or unsupervised learning (Duda and Hart 1973: 45). The distinction is that with *supervised learning* we know the actual status (here, sense label) for each piece of data on which we train, whereas with *unsupervised learning* we do not know the classification of the data in the training sample. Unsupervised learning can thus often be viewed as a *clustering* task (see chapter 14), while supervised learning can usually be seen as a *classification* task (see chapter 16), or equivalently as a function-fitting task where one extrapolates the shape of a function based on some data points.

KNOWLEDGE SOURCES

However, in the Statistical NLP domain, things are often not this simple. Because the production of labeled training data is expensive, people will often want to be able to learn from unlabeled data, but will try to give their algorithms a head start by making use of various *knowledge sources*, such as dictionaries, or more richly structured data, such as aligned bilingual texts. In other methods, the system is seeded with labeled training data, but this data is augmented by further learning from unlabeled data. Rather than trying to force different methods on to a procrustean bed, it usually makes most sense to simply give a precise answer to the question: *What knowledge sources are needed for use of this method?* As we will see, sometimes there are alternative combinations of knowledge sources that can give similar information (e.g., using either aligned bilingual texts, or monolingual texts and a bilingual dictionary).

7.1.2 Pseudowords

In order to test the performance of disambiguation algorithms on a natural ambiguous word, a large number of occurrences has to be disambiguated by hand – a time-intensive and laborious task. In cases like this in which test data are hard to come by, it is often convenient to generate artificial evaluation data for the comparison and improvement of text processing algorithms. In the case of word sense disambiguation these artificial data are called *pseudowords*.

PSEUDOWORDS

Gale et al. (1992e) and Schütze (1992a) show how pseudowords, i.e., artificial ambiguous words, can be created by conflating two or more natural words. For example, to create the pseudoword *banana-door*, one replaces all occurrences of *banana* and *door* in a corpus by the artificial word *banana-door*. Pseudowords make it easy to create large-scale training and test sets for disambiguation while obviating the need for hand-labeling: we regard the text with pseudowords as the ambiguous source text, and the original as the text with the ambiguous words disambiguated.

7.1.3 Upper and lower bounds on performance

While it is important to measure the performance of one's algorithm, numerical evaluation by itself is meaningless without some discussion of how well the algorithm performs *relative to the difficulty of the task.* For example, whereas 90% accuracy is easy to achieve for part-of-speech tagging of English text, it is beyond the capacity of any existing machine translation system. The estimation of upper and lower bounds for the performance of an algorithm is a way to make sense of performance figures (Gale et al. 1992a). It is a good idea for many tasks in NLP, especially if there are no standardized evaluation sets for comparing systems.

UPPER BOUND

The *upper bound* used is usually human performance. In the case of word sense disambiguation, if human judges disagree on the correct sense assignment for a particular context, then we cannot expect an automatic procedure to do better. Determining upper bounds is particularly interesting if the disambiguation algorithm uses a limited representation of contexts, for example just looking at the three words on each side of the ambiguous word. In such a situation, the reason for poor performance may just be that the contextual representations are not very informative so that even humans would not be able to disambiguate very well based on the same information. We can evaluate this by looking at human performance when based on the same limited contextual cues.[1]

An upper bound for word sense disambiguation was established by Gale et al. (1992a). Gale et al. performed tests with the following task: Subjects were given pairs of occurrences and had to decide whether they were instances of the same sense. The task resulted in upper bounds between 97% and 99%. However, most of the words in Gale et al.'s test set have few and clearly distinct senses. In contrast, there are many ambiguous words (in particular, high-frequency ones) that are similar to our example *title*, i.e., their senses are interrelated and overlapping. Inter-judge agreement depends on the type of ambiguity: it is higher for words with clearly distinct senses (95% and higher) and lower for polysemous words with many related senses (perhaps as low as 65% to 70%).[2] The task is also easier when viewed as a yes/no decision task than as an arbitrary clustering task.

This means that we have to look at the properties of an individual ambiguous word to determine whether a disambiguation algorithm does a good job for it. For a word like *bank* we should aim for performance in the ninety percent range, whereas less stringent criteria should be applied to fuzzier cases like *title*, *side*, and *way*.

LOWER BOUND
BASELINE

The *lower bound* or *baseline* is the performance of the simplest possible algorithm, usually the assignment of all contexts to the most frequent sense. A baseline should always be given because raw performance numbers make it impossible to assess how hard disambiguation is for a particular word. An accuracy of 90% is an excellent result for an ambiguous word with two equiprobable senses. The same accuracy for a word with two senses in a 9 to 1 frequency ratio is trivial to achieve – by always selecting the most frequent sense.

▼ Upper and lower bounds are most relevant when we are dealing with a classification task and the evaluation measure is accuracy. Section 8.1 discusses other evaluation measures, in particular, precision and recall.

1. Although, for limited artificial contexts like this, it is of course possible that computers might be able to be more successful than human beings at extracting useful predictive information.

2. See (Jorgensen 1990). To be able to correctly compare the extent of inter-judge agreement across tasks, we need to correct for the expected chance agreement (which depends on the number of senses being distinguished). This is done by the kappa statistic (Siegel and Castellan 1988; Carletta 1996).

Symbol	Meaning
w	an ambiguous word
$s_1, \ldots, s_k, \ldots, s_K$	senses of the ambiguous word w
$c_1, \ldots, c_i, \ldots, c_I$	contexts of w in a corpus
$v_1, \ldots, v_j, \ldots, v_J$	words used as contextual features for disambiguation

Table 7.1 Notational conventions used in this chapter.

7.2 Supervised Disambiguation

In supervised disambiguation, a disambiguated corpus is available for training. There is a training set of exemplars where each occurrence of the ambiguous word w is annotated with a semantic label (usually its contextually appropriate sense s_k). This setting makes supervised disambiguation an instance of statistical classification, the topic of chapter 16. The task is to build a classifier which correctly classifies new cases based on their context of use c_i. This notation, which we will use throughout the remainder of the chapter, is shown in table 7.1.

We have selected two of the many supervised algorithms that have been applied to word sense disambiguation that exemplify two important theoretical approaches in statistical language processing: Bayesian classification (the algorithm proposed by Gale et al. (1992b)) and Information Theory (the algorithm proposed by Brown et al. (1991b)). They also demonstrate that very different sources of information can be employed successfully for disambiguation. The first approach treats the context of occurrence as a bag of words without structure, but it integrates information from many words in the context window. The second approach looks at only one informative feature in the context, which may be sensitive to text structure. But this feature is carefully selected from a large number of potential 'informants.'

7.2.1 Bayesian classification

The idea of the Bayes classifier which we will present for word senses is that it looks at the words around an ambiguous word in a large context window. Each content word contributes potentially useful information about which sense of the ambiguous word is likely to be used with it. The classifier does no feature selection. Instead it combines the evidence from all features. The specific formalization we describe is due to Gale et al. (1992b). The supervised training of the classifier assumes that we have a corpus where each use of ambiguous words is labeled with its correct sense.

BAYES CLASSIFIER
BAYES DECISION RULE

A *Bayes classifier* applies the *Bayes decision rule* when choosing a class, the rule that minimizes the probability of error (Duda and Hart 1973: 10–43):

(7.2) **Bayes decision rule**
Decide s' if $P(s'|c) > P(s_k|c)$ for $s_k \neq s'$

The Bayes decision rule is optimal because it minimizes the probability of error. This is true because for each individual case it chooses the class (or sense) with the highest conditional probability and hence the smallest error rate. The error rate for a sequence of decisions (for example, disambiguating all instances of w in a multi-page text) will therefore also be as small as possible.

BAYES' RULE

We usually do not know the value of $P(s_k|c)$, but we can compute it using *Bayes' rule* as in section 2.1.10:

$$P(s_k|c) = \frac{P(c|s_k)}{P(c)} P(s_k)$$

PRIOR PROBABILITY

POSTERIOR PROBABILITY

$P(s_k)$ is the *prior probability* of sense s_k, the probability that we have an instance of s_k if we do not know anything about the context. $P(s_k)$ is updated with the factor $\frac{P(c|s_k)}{P(c)}$ which incorporates the evidence which we have about the context, and results in the *posterior probability* $P(s_k|c)$.

If all we want to do is choose the correct class, we can simplify the classification task by eliminating $P(c)$ (which is a constant for all senses and hence does not influence what the maximum is). We can also use logs of probabilities to make the computation simpler. Then, we want to assign w to the sense s' where:

(7.3)
$$\begin{aligned} s' &= \arg\max_{s_k} P(s_k|c) \\ &= \arg\max_{s_k} \frac{P(c|s_k)}{P(c)} P(s_k) \\ &= \arg\max_{s_k} P(c|s_k) P(s_k) \\ &= \arg\max_{s_k} [\log P(c|s_k) + \log P(s_k)] \end{aligned}$$

NAIVE BAYES

Gale et al.'s classifier is an instance of a particular kind of Bayes classifier, the *Naive Bayes* classifier. Naive Bayes is widely used in machine learning due to its efficiency and its ability to combine evidence from a large number of features (Mitchell 1997: ch. 6). It is applicable if the state of the world that we base our classification on is described as a series of attributes. In our case, we describe the context of w in terms of the words v_j that occur in the context.

NAIVE BAYES ASSUMPTION

The *Naive Bayes assumption* is that the attributes used for description are all conditionally independent:

(7.4) **Naive Bayes assumption**

$P(c|s_k) = P(\{v_j | v_j \text{ in } c\}|s_k) = \prod_{v_j \text{ in } c} P(v_j|s_k)$

BAG OF WORDS

In our case, the Naive Bayes assumption has two consequences. The first is that all the structure and linear ordering of words within the context is ignored. This is often referred to as a *bag of words* model.[3] The other is that the presence of one word in the bag is independent of another. This is clearly not true. For example, *president* is more likely to occur in a context that contains *election* than in a context that contains *poet*. But, as in many other cases, the simplifying assumption makes it possible to adopt an elegant model that can be quite effective despite its shortcomings. Obviously, the Naive Bayes assumption is inappropriate if there are strong conditional dependencies between attributes. But there is a surprisingly large number of cases in which it does well, partly because the decisions made can still be optimal even if the probability estimates are inaccurate due to feature dependence (Domingos and Pazzani 1997).

With the Naive Bayes assumption, we get the following modified decision rule for classification:

(7.5) **Decision rule for Naive Bayes**

Decide s' if $s' = \arg\max_{s_k} [\log P(s_k) + \sum_{v_j \text{ in } c} \log P(v_j|s_k)]$

$P(v_j|s_k)$ and $P(s_k)$ are computed via Maximum-Likelihood estimation, perhaps with appropriate smoothing, from the labeled training corpus:

$$P(v_j|s_k) = \frac{C(v_j, s_k)}{C(s_k)}$$

3. A bag is like a set, but allows repeated elements (we use 'in' rather than '$\in$' in equation (7.4) because we are treating c as a bag).

```
1  **comment**: Training
2  **for** all senses $s_k$ of $w$ **do**
3      **for** all words $v_j$ in the vocabulary **do**
4          $P(v_j|s_k) = \frac{C(v_j,s_k)}{C(v_j)}$
5      **end**
6  **end**
7  **for** all senses $s_k$ of $w$ **do**
8      $P(s_k) = \frac{C(s_k)}{C(w)}$
9  **end**
10 **comment**: Disambiguation
11 **for** all senses $s_k$ of $w$ **do**
12     $\text{score}(s_k) = \log P(s_k)$
13     **for** all words $v_j$ in the context window $c$ **do**
14         $\text{score}(s_k) = \text{score}(s_k) + \log P(v_j|s_k)$
15     **end**
16 **end**
17 choose $s' = \arg\max_{s_k} \text{score}(s_k)$
```

Figure 7.1 Bayesian disambiguation.

Sense	Clues for sense
medication	*prices*, *prescription*, *patent*, *increase*, *consumer*, *pharmaceutical*
illegal substance	*abuse*, *paraphernalia*, *illict*, *alcohol*, *cocaine*, *traffickers*

Table 7.2 Clues for two senses of *drug* used by a Bayesian classifier. Adapted from (Gale et al. 1992b: 419).

$$P(s_k) = \frac{C(s_k)}{C(w)}$$

where $C(v_j, s_k)$ is the number of occurrences of v_j in a context of sense s_k in the training corpus, $C(s_k)$ is the number of occurrences of s_k in the training corpus, and $C(w)$ is the total number of occurrences of the ambiguous word w. Figure 7.1 summarizes the algorithm.

Gale, Church and Yarowsky (1992b; 1992c) report that a disambiguation system based on this algorithm is correct for about 90% of occurrences for six ambiguous nouns in the Hansard corpus: *duty*, *drug*, *land*, *language*, *position*, and *sentence*.

Table 7.2 gives some examples of words that are good clues for two

Ambiguous word	Indicator	Examples: value → sense
prendre	object	*mesure* → *to take*
		décision → *to make*
vouloir	tense	present → *to want*
		conditional → *to like*
cent	word to the left	*per* → %
		number → *c.* [money]

Table 7.3 Highly informative indicators for three ambiguous French words.

senses of *drug* in the Hansard corpus. For example, *prices* is a good clue for the 'medication' sense. This means that $P(prices|\text{'medication'})$ is large and $P(prices|\text{'illicit substance'})$ is small and has the effect that a context of *drug* containing *prices* will have a higher score for 'medication' and a lower score for 'illegal substance' (as computed on line 14 in figure 7.1).

7.2.2 An information-theoretic approach

The Bayes classifier attempts to use information from all words in the context window to help in the disambiguation decision, at the cost of a somewhat unrealistic independence assumption. The information theoretic algorithm which we turn to now takes the opposite route. It tries to find a single contextual feature that reliably indicates which sense of the ambiguous word is being used. Some of Brown et al.'s (1991b) examples of indicators for French ambiguous words are listed in table 7.3. For the verb *prendre*, its object is a good indicator: *prendre une mesure* translates as *to **take** a measure*, *prendre une décision* as *to **make** a decision*. Similarly, the tense of the verb *vouloir* and the word immediately to the left of *cent* are good indicators for these two words as shown in table 7.3.

FLIP-FLOP ALGORITHM

In order to make good use of an informant, its values need to be categorized as to which sense they indicate, e.g., *mesure* indicates *to take*, *décision* indicates *to make*. Brown et al. use the *Flip-Flop algorithm* for this purpose. Let $t_1, \ldots, t_m$ be the translations of the ambiguous word, and $x_1, \ldots, x_n$ the possible values of the indicator. Figure 7.2 shows the Flip-Flop algorithm for this case. The version of the algorithm described here only disambiguates between two senses. See Brown et al. (1991a) for an extension to more than two senses. Recall the definition of mutual

```
1 find random partition $P = \{P_1, P_2\}$ of $\{t_1, \ldots, t_m\}$
2 while (improving) do
3         find partition $Q = \{Q_1, Q_2\}$ of $\{x_1, \ldots, x_n\}$
4             that maximizes $I(P;\ Q)$
5         find partition $P = \{P_1, P_2\}$ of $\{t_1, \ldots, t_m\}$
7             that maximizes $I(P;\ Q)$
8 end
```

Figure 7.2 The Flip-Flop algorithm applied to finding indicators for disambiguation.

information from section 2.2.3:

$$I(X;\ Y) = \sum_{x \in X} \sum_{y \in Y} \mathrm{p}(x, y) \log \frac{\mathrm{p}(x, y)}{\mathrm{p}(x)\,\mathrm{p}(y)}$$

It can be shown that each iteration of the Flip-Flop algorithm increases the mutual information $I(P;\ Q)$ monotonically, so a natural stopping criterion is that $I(P;\ Q)$ does not increase any more or only insignificantly.

As an example, assume we want to translate *prendre* based on its object and that we have $\{t_1, \ldots, t_m\} = \{$*take*, *make*, *rise*, *speak*$\}$ and $\{x_1, \ldots, x_n\} = \{$*mesure*, *note*, *exemple*, *décision*, *parole*$\}$ (cf. (Brown et al. 1991b: 267)). The initial partition P of the senses might be $P_1 = \{$*take*, *rise*$\}$ and $P_2 = \{$*make*, *speak*$\}$. Which partition Q of the indicator values would give us maximum $I(P;\ Q)$? Obviously, the answer depends on the particular data we are working with. But let us assume that *prendre* is translated by *take* when occurring with the objects *mesure*, *note*, and *exemple* (corresponding to the phrases *take a measure*, *take notes*, and *take an example*), and translated by *make*, *speak*, and *rise* when occurring with *décision*, and *parole* (corresponding to the phrases *make a decision*, *make a speech* and *rise to speak*).

Then the partition that will maximize $I(P;\ Q)$ is $Q_1 = \{$*mesure*, *note*, *exemple*$\}$ and $Q_2 = \{$*décision*, *parole*$\}$ since this division of the indicator values gives us the most information for distinguishing the translations in P_1 from the translations in P_2. We only make an incorrect decision when *prendre la parole* is translated as *rise to speak*, but this cannot be avoided since *rise* and *speak* are in two different partition groups.

The next two steps of the algorithm then repartition P as $P_1 = \{$*take*$\}$ and $P_2 = \{$*make*, *rise*, *speak*$\}$ and Q as before. This partition is always correct for *take*. We would have to consider more than two 'senses' if we also wanted to distinguish between the other translations *make*, *rise* and *speak*.

A simple exhaustive search for the best partition of the French translations and the best possible indicator values would take exponential time. The Flip-Flop algorithm is an efficient linear-time algorithm for computing the best partition of values for a particular indicator, based on the splitting theorem (Breiman et al. 1984). We run the algorithm for all possible indicators and then choose the indicator with the highest mutual information. Brown et al. found that this was the accusative object for *prendre*, tense for *vouloir* and the preceding word for *cent* as shown in table 7.3.

Once an indicator and a particular partition of its values has been determined, disambiguation is simple:

1. For the occurrence of the ambiguous word, determine the value x_i of the indicator.

2. If x_i is in Q_1, assign the occurrence to sense 1, if x_i is in Q_2, assign the occurrence to sense 2.

Brown et al. (1991b) report a 20% improvement in the performance of a machine translation system (from 37 to 45 sentences correct out of 100) when the information-theoretic algorithm is incorporated into the system.

We call the algorithm *supervised* because it requires a labeled training set. However in Brown et al.'s (1991b) work, each occurrence of, say, French *cent* is 'labeled' not with its sense but by its corresponding English translation. These class labels are not the senses. For example, some of the labels of the French word *cent* are (English) *per* and the numbers 0, *one*, 2, and 8. The algorithm groups the labels into two classes, $Q_1 = \{per\}$ and $Q_2 = \{0, one, 2, 8\}$ which are then interpreted as the two senses of *cent*, corresponding to the English translations % (percent sign) and *cent* (with the variants *c.* and *sou*). There is thus a many-to-one mapping from labels to senses.

7.3 Dictionary-Based Disambiguation

If we have no information about the sense categorization of specific instances of a word, we can fall back on a general characterization of the senses. This section describes disambiguation methods that rely on the definition of senses in dictionaries and thesauri. Three different types of information have been used. Lesk (1986) exploits the sense definitions in the dictionary directly. Yarowsky (1992) shows how to apply the semantic categorization of words (derived from the categories in Roget's thesaurus) to the semantic categorization and disambiguation of contexts. In Dagan and Itai's method (1994), translations of the different senses are extracted from a bilingual dictionary and their distribution in a foreign language corpus is analyzed for disambiguation. Finally, we will see how a careful examination of the distributional properties of senses can lead to significant improvements in disambiguation. Commonly, ambiguous words are only used with one sense in any given discourse and with any given collocate (the *one sense per discourse* and *one sense per collocation* hypotheses).

7.3.1 Disambiguation based on sense definitions

Lesk (1986) starts from the simple idea that a word's dictionary definitions are likely to be good indicators for the senses they define.[4] Suppose that two of the definitions of *cone* are as follows:

1. a mass of ovule-bearing or pollen-bearing scales or bracts in trees of the pine family or in cycads that are arranged usually on a somewhat elongated axis,
2. something that resembles a cone in shape: as ... a crisp cone-shaped wafer for holding ice cream.

If either *tree* or *ice* occur in the same context as *cone*, then chances are that the occurrence belongs to the sense whose definition contains that word: sense 1 for *tree*, sense 2 for *ice*.

Let $D_1, \ldots, D_K$ be the dictionary definitions of the senses $s_1, \ldots, s_K$ of the ambiguous word w, represented as the bag of words occurring in the definition, and E_{v_j} the dictionary definition of a word v_j occurring in the context of use c of w, represented as the bag of words occurring in the definition of v_j. (If $s_{j_1}, \ldots, s_{j_L}$ are the senses of v_j, then $E_{v_j} = \bigcup_{j_i} D_{j_i}$. We simply ignore sense distinctions for the words v_j that occur in the context of w.) Then Lesk's algorithm can be described as shown in figure 7.3. For the overlap function, we can just count the number of common words in

4. Lesk credits Margaret Millar and Lawrence Urdang with the original proposal of the algorithm.

```
1 comment: Given: context $c$
2 for all senses $s_k$ of $w$ do
3     score($s_k$) = overlap($D_k, \bigcup_{v_j \text{ in } c} E_{v_j}$)
4 end
5 choose $s'$ s.t. $s' = \arg\max_{s_k} \text{score}(s_k)$
```

Figure 7.3 Lesk's dictionary-based disambiguation algorithm. D_k is the set of words occurring in the dictionary definition of sense s_k. E_{v_j} is the set of words occurring in the dictionary definition of word v_j (that is, the union of all the sense definitions of v_j).

Sense		Definition
s_1	tree	a tree of the olive family
s_2	burned stuff	the solid residue left when combustible material is burned

Table 7.4 Two senses of *ash*.

Scores		Context
s_1	s_2	
0	1	This cigar burns slowly and creates a stiff *ash*.
1	0	The *ash* is one of the last trees to come into leaf.

Table 7.5 Disambiguation of *ash* with Lesk's algorithm. The score is the number of (stemmed) words that are shared by the sense definition and the context. The first sentence is disambiguated as 'burned stuff' because one word is shared with the definition of sense s_2, *burn*, and there are no common words for the other sense. In the second example, the word shared with the definition of s_1 ('tree') is *tree.*

the definition D_k of sense s_k and the union $\bigcup_{v_j \text{ in } c} E_{v_j}$ of the definitions of the words v_j in the context. Or we could use any of the similarity functions which we present in table 8.7.

One of Lesk's examples is the word *ash* with the senses in table 7.4. The two contexts in table 7.5 are correctly disambiguated when scored on the number of words common with the different sense definitions.

By itself, information of this sort derived from a dictionary is insufficient for high quality word sense disambiguation. Lesk reports accuracies between 50% and 70% when the algorithm is applied to a sample of ambiguous words. He suggests various optimizations that could improve performance. For example, one could run several iterations of the algorithm on a text. Instead of using the union of all words E_{v_j} occurring in the definition of v_j, one could only use the words in the definitions of the contextually appropriate senses as determined in the previous iteration of the algorithm. One would hope that the iterated algorithm eventually settles on the correct sense of each word in the text. Pook and Catlett (1988) suggest another improvement: to expand each word in the context with a list of synonyms from a thesaurus. Such an algorithm combines elements of dictionary-based and thesaurus-based disambiguation.

7.3.2 Thesaurus-based disambiguation

Thesaurus-based disambiguation exploits the semantic categorization provided by a thesaurus like Roget's (Roget 1946) or a dictionary with subject categories like Longman's (Procter 1978). The basic inference in thesaurus-based disambiguation is that the semantic categories of the words in a context determine the semantic category of the context as a whole, and that this category in turn determines which word senses are used.

The following simple thesaurus-based algorithm was proposed by Walker (1987: 254). The basic information used is that each word is assigned one or more subject codes in the dictionary. If the word is assigned several subject codes, then we assume that they correspond to the different senses of the word. Let $t(s_k)$ be the subject code of sense s_k of ambiguous word w occurring in context c. Then w can be disambiguated by counting the number of words for which the thesaurus lists $t(s_k)$ as a possible topic. We then choose the sense with the highest count as shown in figure 7.4.

Black (1988: 187) achieved only moderate success when applying Walker's algorithm to a sample of five ambiguous words: accuracies around 50%. However, the test words were difficult and highly ambiguous: *interest*, *point*, *power*, *state* and *terms*.

One problem with the algorithm is that a general categorization of words into topics is often inappropriate for a particular domain. For example, *mouse* may be listed as both a mammal and an electronic device in a thesaurus, but in a computer manual it will rarely be evidence for the thesaurus category 'mammal.' A general topic categorization may also have a problem of coverage. We will not find *Navratilova* in a thesaurus from the 1960s (and we may not find any proper nouns). Yet

```
1 **comment**: Given: context $c$
2 **for** all senses $s_k$ of $w$ **do**
3     $\text{score}(s_k) = \sum_{v_j \text{ in } c} \delta(t(s_k), v_j)$
4 **end**
5 choose $s'$ s.t. $s' = \arg\max_{s_k} \text{score}(s_k)$
```

Figure 7.4 Thesaurus-based disambiguation. $t(s_k)$ is the subject code of sense s_k and $\delta(t(s_k), v_j) = 1$ iff $t(s_k)$ is one of the subject codes of v_j and 0 otherwise. The score is the number of words that are compatible with the subject code of sense s_k.

the occurrence of *Navratilova* is an excellent indicator of the category 'sports.'

The algorithm in figure 7.5 for the adaptation of a topic classification to a corpus was proposed by Yarowsky (1992). The algorithm adds words to a category t_i if they occur more often than chance in the contexts of t_i in the corpus. For example, *Navratilova* will occur more often in sports contexts than in other contexts, so it will be added to the sports category.

Yarowsky's algorithm in figure 7.5 uses the Bayes classifier introduced in section 7.2.1 for both adaptation and disambiguation. First we compute a score for each pair of a context c_i in the corpus and a thesaurus category t_l. For example, context (7.6) would get a high score for the thesaurus category 'sports,' assuming that the thesaurus lists *tennis* as a 'sports' word. In Yarowsky's experiments, a context is simply a 100-word window centered around the ambiguous word.

(7.6) It is amazing that Navratilova, who turned 33 earlier this year, continues to play great tennis.

Making a Naive Bayes assumption, we can compute this score(c_i, t_l) as $\log P(t_l|c_i)$ where $P(t_l|c_i)$ is computed as follows.

$$P(t_l|c_i) = \frac{P(c_i|t_l)}{P(c_i)} P(t_l) = \frac{\prod_{v \text{ in } c_i} P(v|t_l)}{\prod_{v \text{ in } c_i} P(v)} P(t_l) \tag{7.7}$$

We then use a threshold α in line 7 to determine which thesaurus categories are salient in a context. A fairly large value for this threshold should be chosen so that only contexts with good evidence for a category are assigned.

```
1  **comment**: Categorize contexts based on categorization of words
2  **for** all contexts $c_i$ in the corpus **do**
3      **for** all thesaurus categories $t_l$ **do**
4          $\text{score}(c_i, t_l) = \log \frac{P(c_i|t_l)}{P(c_i)} P(t_l)$
5      **end**
6  **end**
7  $t(c_i) = \{t_l | \text{score}(c_i, t_l) > \alpha\}$
8  **comment**: Categorize words based on categorization of contexts
9  **for** all words $v_j$ in the vocabulary **do**
10     $V_j = \{c | v_j \text{ in } c\}$
11 **end**
12 **for** all topics $t_l$ **do**
13     $T_l = \{c | t_l \in t(c)\}$
14 **end**
15 **for** all words $v_j$, all topics $t_l$ **do**
16     $P(v_j|t_l) = |V_j \cap T_l| / \sum_j |V_j \cap T_l|$
17 **end**
18 **for** all topics $t_l$ **do**
19     $P(t_l) = \left(\sum_j |V_j \cap T_l|\right) / \left(\sum_l \sum_j |V_j \cap T_l|\right)$
20 **end**
21 **comment**: Disambiguation
22 **for** all senses $s_k$ of $w$ occurring in $c$ **do**
23     $\text{score}(s_k) = \log P(t(s_k)) + \sum_{v_j \text{ in } c} \log P(v_j | t(s_k))$
24 **end**
25 choose $s'$ s.t. $s' = \arg\max_{s_k} \text{score}(s_k)$
```

Figure 7.5 Adaptive thesaurus-based disambiguation. Yarowsky's algorithms for adapting a semantic categorization of words and for thesaurus-based disambiguation. $P(v_j|t_l)$ on line 16 is estimated as the proportion of contexts of topic t_l that contain word v_j.

Now we can adjust the semantic categorization in the thesaurus to our corpus (represented as the set of contexts $\{c_i\}$). On line 16, we estimate $P(v_j|t_l)$ as the proportion of contexts of v_j that are in category t_l. If v_j is covered in the thesaurus, then this will adapt v_j's semantic categories to the corpus (for example, *stylus* may get a high score as a computer term even though the thesaurus only lists it in the category 'writing'). If v_j is not covered, then it will be added to the appropriate categories (the

Word	Sense	Roget category	Accuracy
bass	musical senses	MUSIC	99%
	fish	ANIMAL, INSECT	100%
star	space object	UNIVERSE	96%
	celebrity	ENTERTAINER	95%
	star shaped object	INSIGNIA	82%
interest	curiosity	REASONING	88%
	advantage	INJUSTICE	34%
	financial	DEBT	90%
	share	PROPERTY	38%

Table 7.6 Some results of thesaurus-based disambiguation. The table shows the senses of three ambiguous words, the Roget categories they correspond to, and the accuracy of the algorithm in figure 7.5. Adapted from (Yarowsky 1992).

case of *Navratilova*). The prior probability of t_l is simply computed as its relative frequency, adjusted for the fact that some contexts will have no semantic categories and others more than one (line 19).

The values $P(v_j|t_l)$ computed on line 16 are then used for disambiguation in analogy to the Bayesian algorithm we discussed earlier (see figure 7.1). Yarowsky (1992) recommends smoothing for some of the maximum likelihood estimates (see chapter 6).

Table 7.6 shows some results from (Yarowsky 1992). The method achieves high accuracy when thesaurus categories and senses align well with topics as in the case of *bass* and *star*. When a sense is spread out over several topics, the algorithm fails. Yarowsky calls these *topic-independent distinctions* between senses. For example, the sense 'advantage' of *interest* (as in *self-interest*) is not topic-specific. Self-interest can occur in music, entertainment, space exploration, finance, etc. Therefore, a topic-based classification does not do well on this sense.

TOPIC-INDEPENDENT DISTINCTIONS

7.3.3 Disambiguation based on translations in a second-language corpus

The third dictionary-based algorithm makes use of word correspondences in a bilingual dictionary (Dagan et al. 1991; Dagan and Itai 1994). We will refer to the language of application (the one for which we want

	Sense 1	Sense 2
Definition	legal share	attention, concern
Translation	*Beteiligung*	*Interesse*
English collocation	*acquire an interest*	*show interest*
Translation	*Beteiligung erwerben*	*Interesse zeigen*

Table 7.7 How to disambiguate *interest* using a second-language corpus.

to do disambiguation) as the *first language* and the target language in the bilingual dictionary as the *second language*. For example, if we want to disambiguate English based on a German corpus, then English is the first language, German is the second language, and we need an English-German dictionary (one with English headwords and German entries).

The basic idea of Dagan and Itai's algorithm is best explained with the example in table 7.7. English *interest* has two senses with two different translations in German. Sense 1 translates as *Beteiligung* (*legal share*, as in "a 50% interest in the company") and Sense 2 translates as *Interesse* (*attention, concern*, as in "her interest in mathematics"). (There are other senses of *interest* which we will ignore here.) In order to disambiguate an occurrence of *interest* in English, we identify the phrase it occurs in and search a German corpus for instances of the phrase. If the phrase occurs with only one of the translations of *interest* in German, then we assign the corresponding sense whenever *interest* is used in this phrase.

As an example, suppose *interest* is used in the phrase *showed interest*. The German translation of *show*, 'zeigen,' will only occur with *Interesse* since "legal shares" are usually not shown. We can conclude that *interest* in the phrase *to show interest* belongs to the sense *attention, concern.* On the other hand, the only frequently occurring translation of the phrase *acquired an interest* is *erwarb eine Beteiligung*, since *interest* in the sense 'attention, concern' is not usually acquired. This tells us that a use of *interest* as the object of *acquire* corresponds to the second sense, "legal share."

A simple implementation of this idea is shown in figure 7.6. For the above example the relation R is 'is-object-of' and the goal would be to disambiguate *interest* in $R(\textit{interest}, \textit{show})$. To do this, we count the number of times that translations of the two senses of *interest* occur with translations of *show* in the second language corpus. The count of $R(\textit{Interesse}, \textit{zeigen})$ would be higher than the count of $R(\textit{Beteiligung}, \textit{zeigen})$, so we

```
1 comment: Given: a context $c$ in which $w$ occurs in relation $R(w, v)$
2 for all senses $s_k$ of $w$ do
3     score$(s_k) = |\{c \in S | \exists w' \in T(s_k), v' \in T(v) : R(w', v') \in c\}|$
4 end
5 choose $s' = \arg\max_{s_k}$ score$(s_k)$
```

Figure 7.6 Disambiguation based on a second-language corpus. S is the second-language corpus, $T(s_k)$ is the set of possible translations of sense s_k, and $T(v)$ is the set of possible translations of v. The score of a sense is the number of times that one of its translations occurs with translations of v in the second-language corpus.

would choose the sense 'attention, concern,' corresponding to *Interesse.*

The algorithm used by Dagan and Itai is more complex: it disambiguates only if a decision can be made reliably. Consider the example of Hebrew *ro'sh* which has two possible English translations, *top* and *head*. Dagan and Itai found 10 examples of the relation *stand at head* and 5 examples of the relation *stand at top* in their English second-language corpus. This suggests that *stand at head* is more likely to translate the Hebrew phrase *'amad be-ro'sh* correctly. However, we can expect "stand at head" to be incorrect in a large proportion of the translations (approximately $\frac{5}{5+10} \approx 0.33$). In many cases, it is better to avoid a decision than to make an error with high probability. In a large system in which each component has a certain error rate, an accuracy of about 0.67 as in the above example is unacceptable. If a sentence passes through five components, each with an error rate of 0.33, then overall system accuracy could be as low as 14%: $(1 - 0.33)^5 \approx 0.14$. Dagan and Itai show how the probability of error can be estimated. They then make decisions only when the level of confidence is 90% or higher.

7.3.4 One sense per discourse, one sense per collocation

The dictionary-based algorithms we have looked at so far process each occurrence separately. But there are constraints between different occurrences that can be exploited for disambiguation. This section discusses work by Yarowsky (1995) which has focussed on two such constraints:

- **One sense per discourse.** The sense of a target word is highly consistent within any given document.

Discourse	Initial label	Context
d_1	living	the existence of *plant* and animal life
	living	classified as either *plant* or animal
	?	Although bacterial and *plant* cells are enclosed
d_2	living	contains a varied *plant* and animal life
	living	the most common *plant* life
	living	slight within Arctic *plant* species
	factory	are protected by *plant* parts remaining from

Table 7.8 Examples of the one sense per discourse constraint. The table shows contexts from two different documents, d_1 and d_2. One context in d_1 lacks sufficient local information for disambiguation ("?"). Local information is misleading for the last context in d_2. The one sense per discourse constraint can be used to counteract lacking or misleading information in such cases. It will correctly assign the unclassified and the misclassified contexts to 'living.' Adapted from (Yarowsky 1995).

- **One sense per collocation.** Nearby words provide strong and consistent clues to the sense of a target word, conditional on relative distance, order and syntactic relationship.

As an example for the first constraint consider the word *plant*. The constraint captures the intuition that if the first occurrence of *plant* is a use of the sense 'living being,' then later occurrences are likely to refer to living beings too. Table 7.8 shows two examples. This constraint is especially usable when the material to be disambiguated is a collection of small documents, or can be divided into short 'discourses' by the kind of method discussed in section 15.5. Then, this simple property of word senses can be used quite effectively as we will see below.

The second constraint makes explicit the basic assumption that most work on statistical disambiguation relies on: that word senses are strongly correlated with certain contextual features like other words in the same phrasal unit. Yarowsky's (1995) approach is similar to Brown et al.'s (1991b) information-theoretic method, which we introduced in section 7.2.2, in that he selects the strongest collocational feature for a particular context and disambiguates based only on this feature. Collocational features are ranked according to the following ratio:

$$\frac{P(s_{k_1}|f)}{P(s_{k_2}|f)} \quad (7.8)$$

which basically is the ratio of the number of occurrences of sense s_{k_1} with collocation f divided by the number of occurrences of sense s_{k_2} with collocation f (again, smoothing is important if the collocation and/or senses occur infrequently, see Yarowsky (1994)).

Relying on only the strongest feature has the advantage that no integration of different sources of evidence is necessary. Many statistical methods, such as the Naive Bayes method used in section 7.2.1 or the dictionary-based methods presented earlier in this section, assume independence when evidence is combined. Since independence rarely holds, it is sometimes better to avoid the need for combining evidence altogether, and to rely on just one reliable piece of evidence. The more complex alternative is to accurately model the dependencies between sources of evidence (see chapter 16).

Figure 7.7 is a schematic description of an algorithm proposed by Yarowsky that combines both constraints. The algorithm iterates building two interdependent sets for each sense s_k. F_k contains characteristic collocations. E_k is the set of contexts of the ambiguous word w that are currently assigned to s_k.

On line 3, F_k is initialized from the dictionary definition of s_k or from another source (for example, a set of collocations entered manually by a lexicographer or a set of collocations from a small hand-labeled training set). E_k is initially empty.

The iteration begins by assigning all contexts with a characteristic collocation from F_k to E_k (line 11). For example, all contexts of *interest* in which *interest* is the object of the verb *show* would be assigned to $E_{\text{'attention, concern'}}$ if "is the object of *show*" is one of the collocations in $F_{\text{'attention, concern'}}$. The set of characteristic collocations is then recomputed by selecting those collocations that are most characteristic of the just updated E_k (line 14).

After this part of the algorithm has been completed, the constraint "one sense per discourse" is applied. All instances of the ambiguous word w are assigned to the majority sense in a document or discourse (line 20). Table 7.8 gave two examples of this process.

Yarowsky demonstrates that this algorithm is highly effective. Different versions achieve between 90.6% and 96.5% accuracy. The error rate is reduced by 27% when the discourse constraint (lines 18–21) is incorporated. This is a surprisingly good performance given that the algorithm does not need a labeled set of training examples.

```
comment: Initialization
for all senses $s_k$ of $w$ do
    $F_k$ = the set of collocations in $s_k$'s dictionary definition
end
for all senses $s_k$ of $w$ do
    $E_k = \emptyset$
end
comment: One sense per collocation
while (at least one $E_k$ changed in the last iteration) do
        for all senses $s_k$ of $w$ do
            $E_k = \{c_i | \exists f_m : f_m \in c_i \wedge f_m \in F_k\}$
        end
        for all senses $s_k$ of $w$ do
            $F_k = \{f_m | \forall n \neq k \frac{P(s_k|f_m)}{P(s_n|f_m)} > \alpha\}$
        end
end
comment: One sense per discourse
for all documents $d_m$ do
    determine the majority sense $s_k$ of $w$ in $d_m$
    assign all occurrences of $w$ in $d_m$ to $s_k$
end
```

Figure 7.7 Disambiguation based on "one sense per collocation" and "one sense per discourse."

7.4 Unsupervised Disambiguation

All that the methods discussed in the last section require for disambiguation are basic lexical resources, a small training set, or a few collocation seeds. Although this seems little to ask for, there are situations in which even such a small amount of information is not available. In particular, this is often the case when dealing with information from specialized domains, for which there may be no available lexical resources.[5] For example, information retrieval systems must be able to deal with text collections from any subject area. General dictionaries are less useful for domain-specific collections. A data base of chemical abstracts mostly

5. However, there are specialized dictionaries in some fields, such as for medical and scientific terms.

contains documents that belong to the category "chemistry" in a generic semantic classification. A generic thesaurus-based disambiguation algorithm would therefore be of little use. One cannot expect the user of an information retrieval system to define the senses of ambiguous words or to provide a training set for a new text collection. With the surge in on-line material in recent years, there is an increasing number of scenarios where outside sources of information are not available for disambiguation.

SENSE TAGGING

Strictly speaking, completely unsupervised disambiguation is not possible if we mean *sense tagging*: an algorithm that labels occurrences as belonging to one sense or another. Sense tagging requires that some characterization of the senses be provided. However, sense *discrimination* can be performed in a completely unsupervised fashion: one can cluster the contexts of an ambiguous word into a number of groups and discriminate between these groups without labeling them. Several such sense discrimination algorithms have been proposed. We will describe one of them here, *context-group discrimination*, largely following Schütze (1998).[6] Note also the similarity to Brown et al.'s approach described in section 7.2.2. Brown et al. (1991b) cluster *translations* of an ambiguous word, which can be thought of as a type of prelabeling of the occurrences of the ambiguous word w. Here, we will look at a completely unsupervised algorithm that clusters unlabeled occurrences.

CONTEXT-GROUP DISCRIMINATION

The probabilistic model is the same as that developed by Gale et al. (section 7.2.1). For an ambiguous word w with senses $s_1, \ldots, s_k, \ldots, s_K$, we estimate the conditional probability of each word v_j occurring in a context where w is being used in a particular sense s_k, that is, $P(v_j|s_k)$.

In contrast to Gale et al.'s Bayes classifier, parameter estimation in unsupervised disambiguation is not based on a labeled training set. Instead, we start with a random initialization of the parameters $P(v_j|s_k)$. The $P(v_j|s_k)$ are then reestimated by the EM algorithm (see section 14.2.2). After the random initialization, we compute for each context c_i of w the probability $P(c_i|s_k)$ that it was generated by sense s_k. We can use this preliminary categorization of the contexts as our training data and then reestimate the parameters $P(v_j|s_k)$ so as to maximize the likelihood of the data given the model. The algorithm is developed in figure 7.8.

The EM algorithm is guaranteed to increase the log likelihood of the

6. For consistency we reuse the probabilistic model introduced in section 7.2.1 and section 7.3.2, instead of Schütze's.

1. **Initialize** the parameters of the model μ randomly. The parameters are $P(v_j|s_k)$, $1 \leq j \leq J, 1 \leq k \leq K$, and $P(s_k)$, $1 \leq k \leq K$.

   Compute the log of the likelihood of the corpus C given the model μ as the product of the probabilities $P(c_i)$ of the individual contexts c_i (where $P(c_i) = \sum_{k=1}^{K} P(c_i|s_k)P(s_k)$):

$$l(C|\mu) = \log \prod_{i=1}^{I} \sum_{k=1}^{K} P(c_i|s_k)P(s_k) = \sum_{i=1}^{I} \log \sum_{k=1}^{K} P(c_i|s_k)P(s_k)$$

2. While $l(C|\mu)$ is improving repeat:

   (a) **E-step.** For $1 \leq k \leq K, 1 \leq i \leq I$ estimate h_{ik}, the posterior probability that s_k generated c_i, as follows:

$$h_{ik} = \frac{P(c_i|s_k)}{\sum_{k=1}^{K} P(c_i|s_k)}$$

   To compute $P(c_i|s_k)$, we make the by now familiar Naive Bayes assumption:

$$P(c_i|s_k) = \prod_{v_j \in c_i} P(v_j|s_k)$$

   (b) **M-step. Re-estimate** the parameters $P(v_j|s_k)$ and $P(s_k)$ by way of maximum likelihood estimation:

$$P(v_j|s_k) = \frac{\sum_{i=1}^{I} \sum_{\{c_i: v_j \in c_i\}} h_{ik}}{Z_j}$$

   where $\sum_{\{c_i: v_j \in c_i\}}$ sums over all contexts in which v_j occurs and $Z_j = \sum_{k=1}^{K} \sum_{i=1}^{I} \sum_{\{c_i: v_j \in c_i\}} h_{ik}$ is a normalizing constant.
   Recompute the probabilities of the senses as follows:

$$P(s_k) = \frac{\sum_{i=1}^{I} h_{ik}}{\sum_{k=1}^{K} \sum_{i=1}^{I} h_{ik}} = \frac{\sum_{i=1}^{I} h_{ik}}{I}$$

Figure 7.8 An EM algorithm for learning a word sense clustering. K is the number of desired senses; $c_1, \ldots, c_i, \ldots, c_I$ are the contexts of the ambiguous word in the corpus; and $v_1, \ldots, v_j, \ldots, v_J$ are the words being used as disambiguating features.

model given the data in each step. Therefore, the stopping criterion for the algorithm is to stop when the likelihood (computed in step 1) is no longer increasing significantly.

Once the parameters of the model have been estimated, we can disambiguate contexts of w by computing the probability of each of the senses based on the words v_j occurring in the context. Again, we make the Naive Bayes assumption and use the Bayes decision rule (7.5):

$$\text{Decide } s' \text{ if } s' = \arg\max_{s_k} [\log P(s_k) + \sum_{v_j \in c} \log P(v_j | s_k)]$$

The granularity of the sense classification of an ambiguous word can be chosen by running the algorithm for a range of values for K, the number of senses. The more senses there are, the more structure the model has, and therefore it will be able to explain the data better. As a result the best possible log likelihood of the model given the data will be higher with each new sense added. However, one can examine by how much the log likelihood increases with each new sense. If it increases strongly because the new sense explains an important part of the data, then this suggests that the new number of senses is justified. If the log likelihood increases only moderately, then the new sense only picks up random variation in the data and it is probably not justified.[7]

A simpler way to determine the number of senses is to make it dependent on how much training material is available. This approach is justified for an information retrieval application by Schütze and Pedersen (1995).

An advantage of unsupervised disambiguation is that it can be easily adapted to produce distinctions between usage types that are more fine-grained than would be found in a dictionary. Again, information retrieval is an application for which this is useful. The distinction between physical banks in the context of bank robberies and banks as abstract corporations in the context of corporate mergers can be highly relevant even if it is not reflected in dictionaries.

If the unsupervised algorithm is run for a large number of senses, say $K = 20$, then it will split dictionary senses into fine-grained contextual variants. For example, the sense ‘lawsuit’ of *suit* could be split into ‘civil suit,’ ‘criminal suit,’ etc. Usually, the induced clusters do not line up well with dictionary senses. Infrequent senses and senses that have few

7. One could choose the optimal number of senses automatically by testing on validation data, as discussed in chapter 6.

Word	Sense	Accuracy	
		μ	σ
suit	lawsuit	95	0
	the suit you wear	96	0
motion	physical movement	85	1
	proposal for action	88	13
train	line of railroad cars	79	19
	to teach	55	31

Table 7.9 Some results of unsupervised disambiguation. The table shows the mean μ and standard deviation σ for ten experiments with different initial conditions for the EM algorithm. Data are from (Schütze 1998: 110).

collocations are hard to isolate in unsupervised disambiguation. Senses like the use of *suit* in the sense 'to be appropriate for' as in *This suits me fine* are unlikely to be discovered. However, such hard to identify senses often carry less content than senses that are tied to a particular subject area. For an information retrieval system, it is probably more important to make the distinction between usage types like 'civil suit' vs. 'criminal suit' than to isolate the verbal sense 'to suit.'

Some results of unsupervised disambiguation are shown in table 7.9. We need to take into account the variability that is due to different initializations here (Step 1 in figure 7.8). The table shows both the average accuracy and the standard deviation over ten trials. For senses with a clear correspondence to a particular topic, the algorithm works well and variability is low. The word *suit* is an example. But the algorithm fails for words whose senses are topic-independent such as 'to teach' for *train* – this failure is not unlike other methods that work with topic information only. In addition to the low average performance, variability is also quite high for topic-independent senses. In general, performance is 5% to 10% lower than that of some of the dictionary-based algorithms as one would expect given that no lexical resources for training or defining senses are used.

7.5 What Is a Word Sense?

Now that we have looked at a wide range of different approaches to word sense disambiguation, let us revisit the question of what precisely a word sense is. It would seem natural to define senses as the mental representations of different meanings of a word. But given how little is known about the mental representation of meaning, it is hard to design experiments that determine how senses are represented by a subject. Some studies ask subjects to cluster contexts. The subject is given a pile of index cards, each with a sentence containing the ambiguous word, and instructions to sort the pile into coherent subgroups. While these experiments have provided many insights (for example, for research on the notion of semantic similarity, see Miller and Charles (1991)), it is not clear how well they model the use of words and senses in actual language comprehension and production. Determining linguistic similarity is not a task that people are confronted with in natural situations. Agreement between clusterings performed by different subjects is low (Jorgensen 1990).

Another problem with many psychological experiments on ambiguity is that they rely on introspection and whatever folk meaning a subject assumes for the word 'sense.' It is not clear that introspection is a valid methodology for getting at the true mental representations of senses since it fails to elucidate many other phenomena. For example, people tend to rationalize non-rational economic decisions (Kahneman et al. 1982).

The most frequently used methodology is to adopt the sense definitions in a dictionary and then to ask subjects to label instances in a corpus based on these definitions. There are different opinions on how well this technique works. Some researchers have reported high agreement between judges (Gale et al. 1992a) as we discussed above. High average

SKEWED DISTRIBUTION

agreement is likely if there are many ambiguous words with a skewed distribution, that is, one sense that is used in most of the occurrences. Sanderson and van Rijsbergen (1998) argue that such skewed distributions are typical of ambiguous words.

However, randomly selecting ambiguous words as was done in (Gale et al. 1992a) introduces a bias which means that their figures may not reflect actual inter-judge agreement. Many ambiguous words with the highest disagreement rates are high-frequency words. So on a per-token basis inter-judge disagreement can be high even if it is lower on a per-type basis. In a recent experiment, Jean Véronis (p.c., 1998) found that there was not a single instance of the frequent French words *correct*, *historique*, *économie*, and *comprendre* with complete agreement among judges. The main reasons Véronis found for inter-judge disagreement were vague dictionary definitions and true ambiguity in the corpus.

Can we write dictionaries that are less vague? Fillmore and Atkins (1994) discuss such issues from a lexicographic perspective. Some authors argue that it is an inherent property of word meaning that several senses of a word can be used simultaneously or *co-activated* (Kilgarriff 1993; Schütze 1997; Kilgarriff 1997), which entails high rates of inter-judge disagreement. Of course, there are puns like (7.9) in which multiple senses are used in a way that seems so special that it would be acceptable for an NLP system to fail:

CO-ACTIVATION

(7.9) In AI, much of the I is in the beholder.

But Kilgarriff (1993) argues that such simultaneous uses of senses are quite frequent in ordinary language. An example is (7.10) where arguably two senses of *competition* are invoked: 'the act of competing' and 'the competitors.'

(7.10) For better or for worse, this would bring competition to the licensed trade.

SYSTEMATIC POLYSEMY

Many cases of 'coactivation' are cases of *systematic polysemy*, lexico-semantic rules that apply to a class of words and systematically change or extend their meaning. (See (Apresjan 1974), (Pustejovsky 1991), (Lakoff 1987), (Ostler and Atkins 1992), (Nunberg and Zaenen 1992), and (Copestake and Briscoe 1995) for theoretical work on systematic polysemy and (Buitelaar 1998) for a recent computational study.) The word *competition* is a case in point. A large number of English words have the same meaning alternation between 'the act of *X*' vs. 'the people doing *X*'. For example, *organization*, *administration*, and *formation* also exhibit it.

A different type of systematic ambiguity that cannot be neglected in practice is that almost all words can also be used as proper nouns, some of them frequently. Examples are *Brown*, *Bush*, and *Army*.

One response to low inter-judge agreement and the low performance of disambiguation algorithms for highly ambiguous words is to only consider coarse-grained distinctions, for example only those that manifest themselves across languages (Resnik and Yarowsky 1998). Systematic polysemy is likely to be similar in many languages, so we would not distinguish the two related senses of *competition* ('the act of competing' and 'the competitors') even if a monolingual dictionary lists them as different. This strategy is similar to ones used in other areas of NLP, such as parsing, where one defines an easier problem, shallow parsing, and does not attempt to solve the hardest problem, the resolution of attachment ambiguities.

Clustering approaches to word sense disambiguation (such as context-group disambiguation) adopt the same strategy. By definition, automatic clustering will only find groups of usages that can be successfully distinguished. This amounts to a restriction to a subpart of the problem that can be solved. Such solutions with a limited scope can be quite useful. Many translation ambiguities are coarse, so that a system restricted to coarse sense distinctions is sufficient. Context-group disambiguation has been successfully applied to information retrieval (Schütze and Pedersen 1995).

Such application-oriented notions of sense have the advantage that it is easy to evaluate them as long as the application that disambiguation is embedded in can be evaluated (for example, translation accuracy for machine translation, the measures of recall and precision – introduced in chapter 8 – for information retrieval). Direct evaluation of disambiguation accuracy and comparison of different algorithms is more difficult, but will be easier in the future with the development of standard evaluation sets. See Mooney (1996) for a comparative evaluation of a number of machine learning algorithms and Towell and Voorhees (1998) for the evaluation of a disambiguator for three highly ambiguous words (*hard*, *serve*, and *line*). A systematic evaluation of algorithms was undertaken as part of the *Senseval* project (unfortunately, after the writing of this chapter). See the website.

SENSEVAL

Another factor that influences what notion of sense is assumed, albeit implicitly, is the *type of information* that is used in disambiguation: co-occurrence (the bag-of-words model), relational information (subject, object, etc.), other grammatical information (such as part-of-speech), collocations (one sense per collocation) and discourse (one sense per discourse). For example, if only co-occurrence information is used, then only 'topical' sense distinctions are recognized, senses that are associated with different domains. The inadequacy of the bag-of-words model for many sense distinctions has been emphasized by Justeson and Katz (1995a). Leacock et al. (1998) look at the combination of topical and collocational information and achieve optimal results when both are used. Choueka and Lusignan (1985) show that humans do surprisingly well at sense discrimination if only a few words of adjacent context are shown – giving more context contributes little to human disambiguation performance. However, that does not necessarily mean that wider context is useless for the computer. Gale et al. (1992b) show that there is additional useful information in the context out to about 50 words on either side of the ambiguous word (using their algorithm), and that there is detectable information about sense distinctions out to a very large distance (thousands of words).

Different types of information may be appropriate to different degrees for different parts of speech. Verbs are best disambiguated by their arguments (subjects and objects), which implies the importance of local information. Many nouns have topically distinct word senses (like *suit* and *bank*) so that a wider context is more likely to be helpful.

Much research remains to be done on word sense disambiguation. In particular, it will become necessary to evaluate algorithms on a representative sample of ambiguous words, an effort few researchers have made so far. Only with more thorough evaluation will it be possible to fully understand the strengths and weaknesses of the disambiguation algorithms introduced in this chapter.

7.6 Further Reading

An excellent recent discussion of both statistical and non-statistical work on word sense disambiguation is (Ide and Véronis 1998). See also (Guthrie et al. 1996). An interesting variation of word sense disambiguation is *sentence boundary identification* (section 4.2.4). The problem is that periods in text can be used either to mark an abbreviation or to mark the end of a sentence. Palmer and Hearst (1997) show how the problem can be cast as the task of disambiguating two 'senses' of the period: ending an abbreviation vs. ending a sentence or both.

SENTENCE BOUNDARY IDENTIFICATION

The common thread in this chapter has been the amount and type of lexical resources used by different approaches. In these remarks, we will first mention a few other methods that fit under the rubrics of supervised, dictionary-based, and unsupervised disambiguation, and then work that did not fit well into our organization of the chapter.

Two important supervised disambiguation methods are k nearest neighbors (kNN), also called memory-based learning (see page 295) and loglinear models. A nearest neighbor disambiguator is introduced in (Dagan et al. 1994, 1997b). The authors stress the benefits of kNN approaches for sparse data. See also (Ng and Lee 1996) and (Zavrel and Daelemans 1997). Decomposable models, a type of loglinear model, can be viewed as a generalization of Naive Bayes. Instead of treating all features as independent, features are grouped into mutually dependent subsets. Independence is then assumed only between features in different subsets, not for all pairs of features as is the case in the Naive Bayes classifier. Bruce and Wiebe (1994) apply decomposable models to disambiguation with good results.

Other disambiguation algorithms that rely on lexical resources are (Karov and Edelman 1998), (Guthrie et al. 1991), and (Dini et al. 1998). Karov and Edelman (1998) present a formalism that takes advantage of evidence both from a corpus and a dictionary, with good disambiguation results. Guthrie et al. (1991) use the subject field codes in (Procter 1978) in a way similar to the thesaurus classes in (Yarowsky 1992). Dini et al. (1998) apply transformation-based learning (see section 10.4.1) to tag ambiguous words with thesaurus categories.

Papers that use clustering include (Pereira et al. 1993; Zernik 1991b; Dolan 1994; Pedersen and Bruce 1997; Chen and Chang 1998). Pereira et al. (1993) cluster contexts of words in a way similar to Schütze (1998), but based on a different formalization of clustering. They do not directly describe a disambiguation algorithm based on the clustering result, but since in this type of unsupervised method assignment to clusters is equivalent to disambiguation, this would be a straightforward extension. See section 14.1.4 for the clustering algorithm they use. Chen and Chang (1998) and Dolan (1994) are concerned with constructing representations for senses by combining several subsenses into one 'supersense.' This type of clustering of subsenses is useful for constructing senses that are coarser than those a dictionary may provide and for relating sense definitions between two dictionaries.

An important issue that comes up in many different approaches to disambiguation is how to combine different types of evidence (McRoy 1992). See (Cottrell 1989; Hearst 1991; Alshawi and Carter 1994; Wilks and Stevenson 1998) for different proposals.

Although we only cover statistical approaches here, work on word sense disambiguation has a long tradition in Artificial Intelligence and Computational Linguistics. Two often-cited contributions are (Kelly and Stone 1975), a hand-constructed rule-based disambiguator, and (Hirst 1987), who exploits selectional restrictions for disambiguation. An excellent overview of non-statistical work on disambiguation can be found in the above-mentioned (Ide and Véronis 1998).

7.7 Exercises

Exercise 7.1 [⋆]

The lower bound of disambiguation accuracy depends on how much information is available. Describe a situation in which the lower bound could be lower than the performance that results from classifying all occurrences of a word as instances of its most frequent sense. (Hint: What knowledge is needed to calculate that lower bound?)

Exercise 7.2 [⋆⋆]

Supervised word sense disambiguation algorithms are quite easy to devise and train. Either implement one of the models discussed above, or design your own and implement it. How good is the performance? Training data are available from the Linguistic Data Consortium (the DSO corpus) and from the WordNet project (semcor). See the website for links to both.

Exercise 7.3 [⋆⋆]

Create an artificial training and test set using pseudowords. Evaluate one of the supervised algorithms on it.

Exercise 7.4 [⋆⋆]

Download a version of Roget's thesaurus from the web (see the website), and implement and evaluate a thesaurus-based algorithm.

Exercise 7.5 [⋆⋆]

The two supervised methods differ on two different dimensions: the number of features used (one vs. many) and the mathematical methodology (information theory vs. Bayesian classification). How would one design a Bayes classifier that uses only one feature and an information-theoretic method that uses many features?

Exercise 7.6 [⋆⋆]

In light of the discussion on closely related and 'co-activated' senses, discuss to what extent pseudowords model ambiguity well.

Exercise 7.7 [⋆⋆]

Lesk's algorithm counts how many words are shared between sense definition and context. This is not optimal since reliance on "non-descript" or stop words like *try* or *especially* can result in misclassifications. Try to come up with refinements of Lesk's algorithm that would weight words according to their expected value in discrimination.

Exercise 7.8 [⋆]

Two approaches use only one feature: information-theoretic disambiguation and Yarowsky's (1995) algorithm. Discuss differences and other similarities between the two approaches.

Exercise 7.9 [⋆]

Discuss the validity of the "one sense per discourse" constraint for different types of ambiguity (types of usages, homonyms etc.). Construct examples where the constraint is expected to do well and examples where it is expected to do poorly.

Exercise 7.10 [⋆⋆]

Evaluate the one sense per discourse constraint on a corpus. Find sections or articles with multiple uses of an ambiguous word, and work out how often they have different senses.

Exercise 7.11 [⋆]

The section on unsupervised disambiguation describes criteria for determining the number of senses of an ambiguous word. Can you think of other criteria? Assume (a) that a dictionary is available (but the word is not listed in it); (b) that a thesaurus is available (but the word is not listed in it).

Exercise 7.12 [⋆]

For a pair of languages that you are familiar with, find three cases of an ambiguous word in the first language for which the senses translate into different words and three cases of an ambiguous words for which at least two senses translate to the same word.

Exercise 7.13 [⋆]

Is it important to evaluate unsupervised disambiguation on a separate test set or does the unsupervised nature of the method make a distinction between training and test set unnecessary? (Hint: It can be important to have a separate test set. Why? See (Schütze 1998: 108).)

Exercise 7.14 [⋆]

Several of the senses of *title* discussed in the beginning of the chapter are related by systematic polysemy. Find other words with the same systematic polysemy.

Exercise 7.15 [⋆⋆]

Pick one of the disambiguation algorithms and apply it to sentence boundary identification.

"There is one, yt is called in the Malaca tongue Durion, and is so good that . . . it doth exceede in savour all others that euer they had seene, or tasted."

(Parke tr. Mendoza's Hist. China 393, 1588)