Similarity-based Word Sense Disambiguation

Yael Karov
Weizmann Institute

Shimon Edelman
Weizmann Institute

We describe a method for automatic word sense disambiguation using a text corpus and a machine-readable dictionary (MRD). The method is based on word similarity and context similarity measures. Words are considered similar if they appear in similar contexts; contexts are similar if they contain similar words. The circularity of this definition is resolved by an iterative, converging process, in which the system learns from the corpus a set of typical usages for each of the senses of the polysemous word listed in the MRD. A new instance of a polysemous word is assigned the sense associated with the typical usage most similar to its context. Experiments show that this method performs well, and can learn even from very sparse training data.

Introduction

Word Sense Disambiguation (WSD) is the problem of assigning a sense to an ambiguous word, using its context. We assume that different senses of a word correspond to different entries in its dictionary definition. For example, `suit` has two senses listed in a dictionary: *an action in court*, and *suit of clothes*. Given the sentence *The union's lawyers are reviewing the suit*, we would like the system to decide automatically that `suit` is used there in its court-related sense (we assume that the part of speech of the polysemous word is known).

In recent years, text corpora have been the main source of information for learning automatic WSD (see, e.g., (Gale, Church, and Yarowsky, 1992)). A typical corpus-based algorithm constructs a training set from all contexts of a polysemous word W in the corpus, and uses it to learn a classifier that maps instances of W (each supplied with its context) into the senses. Because learning requires that the examples in the training set be partitioned into the different senses, and because sense information is not available in the corpus explicitly, this approach depends critically on manual sense tagging — a laborious and time-consuming process that has to be repeated for every word, in every language, and, more likely than not, for every topic of discourse or source of information.

The need for tagged examples creates a problem referred to in previous works as the *knowledge acquisition bottleneck*: training a disambiguator for W requires that the examples in the corpus be partitioned into senses, which, in turn, requires a fully operational disambiguator. The method we propose circumvents this problem by automatically tagging the training set examples for W using other examples, that do not contain W, but do contain related words extracted from its dictionary definition. For instance, in the training set for `suit`, we would use, in addition to the contexts of `suit`, all the contexts of `court` and of `clothes` in the corpus, because `court` and `clothes` appear in the MRD entry of `suit` that defines its two senses. Note that, unlike the contexts of `suit`, which may discuss either court action or clothing, the contexts of `court` are not likely to be especially related to clothing, and, similarly, those of `clothes` will normally have little to do with lawsuits. We will use this observation to tag the original contexts of `suit`.

Another problem that affects the corpus-based WSD methods is the *sparseness of data*: these methods typically rely on the statistics of cooccurrences of words, while many of the possible cooccurrences are not observed even in a very large corpus (Church and Mercer, 1993). We address this problem in several ways. First, instead of tallying word statistics for the examples of each sense (which may be unreliable when the examples are few), we collect sentence-level statistics, representing each sentence by the set of features it contains (more on features in section 3.2). Second, we define a similarity measure on the feature space, which allows us to pool the statistics of similar features. Third, in addition to the examples of the polysemous word W in the corpus, we learn also from the examples of all the words in the dictionary definition of W. In our experiments, this resulted in a training set that could be up to 20 times larger than the set of original examples.

The rest of this paper is organized as follows. Section 1 describes the approach we have developed. In section 2, we report the results of tests we have conducted on the Treebank-2 corpus. Section 3 concludes with a discussion of related methods and a summary. Proofs and other details of our scheme can be found in the appendix.

1. Similarity-based disambiguation

Our aim is to have the system learn to disambiguate the appearances of a polysemous word W with senses $\mathrm{s}_1, \ldots, \mathrm{s}_k$, using the appearances of W in an untagged corpus as examples. To avoid the need to tag the training examples manually, we augment the training set by additional sense-related examples, which we call a *feedback set*. The feedback set for sense s_i of word W is the union of all contexts that contain some noun found in the entry of $\mathrm{s}_i(\mathcal{W})$ in a MRD[1] (high-frequency nouns, and nouns in the intersection of any two sense entries, as well as examples in the intersection of two feedback sets, are discarded). The feedback sets can be augmented, in turn, by original training-set sentences that are closely related (in a sense defined below) to one of the feedback set sentences; these additional examples can then attract other original examples.

The feedback sets constitute a rich source of data that are known to be sorted by sense. Specifically, the feedback set of s_i is known to be more closely related to s_i than to the other senses of the same word. We rely on this observation to tag automatically the examples of W, as follows. Each original sentence containing W is assigned the sense of its most similar sentence in the feedback sets. Two sentences are considered to be similar insofar as they contain similar words (they do not have to share any word);

[1] By MRD we mean a machine-readable dictionary or a thesaurus, or any combination of such knowledge sources.

words are considered to be similar if they appear in similar sentences. The circularity of this definition is resolved by an iterative, converging process, described below.

1.1 Terminology

A *context*, or *example* of the target word W is any sentence that contains W, and (optionally) the two adjacent sentences in the corpus. The *features* of a sentence are its nouns, verbs, and the adjectives of W and of the nouns from W's MRD definition, all used after stemming (it is also possible to use other types of features, such as word n-grams or syntactic information; see section 3.2). As the number of features in the training data can be very large, we automatically assign each relevant feature a weight indicating the extent to which it is indicative of the sense (see section A.3). Features that appear less than two times, and features whose weight falls under a certain threshold are excluded. A sentence is represented by the set of the remaining relevant features it contains.

1.2 Computation of similarity

Our method hinges on the possibility to compute similarity between the original contexts of W and the sentences in the feedback sets. We concentrate on similarities in the way sentences use W, and not in their meaning. Thus, similar words tend to appear in similar contexts, and their textual proximity to the ambiguous word W is indicative of the sense of W. Note that contextually similar words do not have to be synonyms, or to belong to the same lexical category. For example, we consider the words *doctor* and *health* to be similar because they frequently share contexts, although they are far removed from each other in a typical semantic hierarchy such as the WordNet (Miller et al., 1993). Note, further, that because we learn similarity from the training set of W, and not from the entire corpus, it tends to capture regularities with respect to the usage of W, rather than abstract or general regularities. For example, the otherwise unrelated words *war* and *trafficking* are similar in the contexts of the polysemous word *drug* (*narcotic / medicine*), because the expressions *drug trafficking* and *the war on drugs* appear in related contexts of *drug*. As a result, both *war* and *trafficking* are similar in being strongly indicative of the *narcotic* sense of *drug*.

Words and sentences play complementary roles in our approach: a sentence is represented by the set of words it contains, and a word — by the set of sentences in which it appears. Sentences are similar to the extent they contain similar words; words are similar to the extent they appear in similar sentences. Although this definition is circular, it turns out to be of great use, if applied iteratively, as described below.

In each iteration, we update a word similarity matrix $M^{(w)}$, whose rows and columns are labeled by all the words encountered in the training set of W. In that matrix, the cell $M^{(w)}(i,j)$ holds a value between 0 and 1, indicating the extent to which word i is contextually similar to word j. In addition, we keep and update a separate sentence similarity matrix $M_i^{(s)}$ for each sense s$_i$ of W (including a matrix $M_0^{(s)}$ that contains the similarities of the original examples to themselves). The rows in a sentence matrix $M_i^{(s)}$ correspond to the original examples of W, and the columns — to the original examples of W for $i = 0$, and to the feedback-set examples for sense s$_i$, for $i > 0$.

To compute the similarities, we initialize the word similarity matrix to the identity matrix (each word is fully similar to itself, and completely dissimilar to other words), and iterate (see Figure 1):

1. update the sentence similarity matrices $M_i^{(s)}$, using the word similarity matrix $M^{(w)}$;

2. update the word similarity matrix $M^{(w)}$, using the sentence similarity matrices $M_i^{(s)}$.

until the changes in the similarity values are small enough (see section A.1 for a detailed description of the stopping conditions; a proof of convergence appears in the appendix).

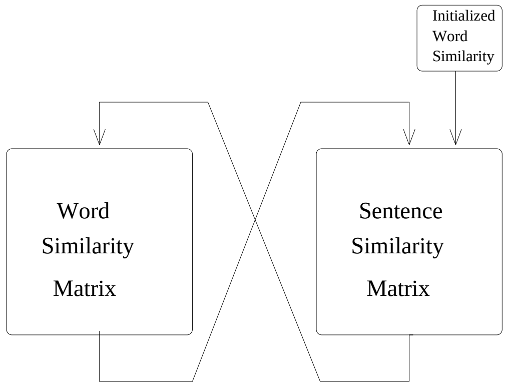

Figure 1
Iterative computation of word and sentence similarities.

1.2.1 The affinity formula. The algorithm for updating the similarity matrices involves an auxiliary relation between words and sentences, which we call *affinity*, introduced to simplify the symmetric iterative treatment of similarity between words and sentences. A word W is assumed to have a certain affinity to every sentence. Affinity (a real number between 0 and 1) reflects the contextual relationships between W and the words of the sentence. If W belongs to a sentence S, its affinity to S is 1; if W is totally unrelated to S, the affinity is close to 0 (this is the most common case); if W is contextually similar to the words of S, its affinity to S is between 0 and 1. In a symmetric manner, a sentence S has some affinity to every word, reflecting the similarity of S to sentences involving that word.

We say that a word *belongs* to a sentence, denoted as $\mathcal{W} \in \mathcal{S}$, if it textually contained there; in this case, sentence is said to *include* the word: $\mathcal{S} \ni \mathcal{W}$. Affinity is then defined as follows:

$$\mathrm{aff}_n(\mathcal{W},\ \mathcal{S}) = \max_{\mathcal{W}_i \in \mathcal{S}} \mathrm{sim}_n(\mathcal{W},\ \mathcal{W}_i) \quad (1)$$

$$\mathrm{aff}_n(\mathcal{S},\ \mathcal{W}) = \max_{\mathcal{S}_j \ni \mathcal{W}} \mathrm{sim}_n(\mathcal{S},\ \mathcal{S}_j) \quad (2)$$

where n denotes the iteration number.[2] The initial representation of a sentence, as the set of words that it directly contains, is now augmented by a similarity-based representation; The sentence contains more information or features than the words directly contained in it. Every word has some affinity to the sentence, and the sentence can be represented by a vector indicating the affinity of each word to it. Similarly, every word can be represented by the affinity of every sentence to it. Note that affinity is asymmetric: aff$(\mathcal{S},\ \mathcal{W}) \neq$ aff$(\mathcal{W},\ \mathcal{S})$, because W may be similar to one of the words in S, which, however, is not one of the topic words of S; it is not an important word in s. In this case, aff$(\mathcal{W},\ \mathcal{S})$ is high, because W is similar to a word in S, but aff$(\mathcal{S},\ \mathcal{W})$ is low, because S is not a representative example of the usage of the word W.

1.2.2 The similarity formula. We define the similarity of $\mathcal{W}_1$ to $\mathcal{W}_2$ to be the average affinity of sentences that include $\mathcal{W}_1$ to those that include $\mathcal{W}_2$. The similarity of a sentence $\mathcal{S}_1$ to another sentence $\mathcal{S}_2$ is a weighted average of the affinity of the words in $\mathcal{S}_1$ to those in $\mathcal{S}_2$:

$$\begin{aligned} \mathrm{sim}_{n+1}(\mathcal{S}_1,\ \mathcal{S}_2) &= \sum_{\mathcal{W}\in\mathcal{S}_1} \mathrm{weight}(\mathcal{W},\ \mathcal{S}_1)\cdot \mathrm{aff}_n(\mathcal{W},\ \mathcal{S}_2) \qquad (3)\\ \mathrm{sim}_{n+1}(\mathcal{W}_1,\ \mathcal{W}_2) &= \sum_{\mathcal{S}\ni\mathcal{W}_1} \mathrm{weight}(\mathcal{S},\ \mathcal{W}_1)\cdot \mathrm{aff}_n(\mathcal{S},\ \mathcal{W}_2) \qquad (4)\end{aligned}$$

where the weights sum to 1.[3]

1.2.3 The importance of iteration. Initially, only identical words are considered similar, so that aff$(\mathcal{W},\ \mathcal{S}) = 1$ if $\mathcal{W} \in \mathcal{S}$; the affinity is zero otherwise. Thus, in the first iteration,

[2] At a first glance it may seem that the mean rather than the maximal similarity of W to the words of a sentence should determine the affinity between the two. However, any definition of affinity that takes into account more words than just the one with the maximal similarity to W, may result in a word being directly contained in the sentence, but having an affinity to it that is smaller than 1.

[3] The weight of a word estimates its expected contribution to the disambiguation task, and is a product of several factors: the frequency of the word in the corpus, its frequency in the training set relative to that in the entire corpus; the textual distance from the target word, and its part of speech (more details on word weights appear in section A.3). All the sentences that include a given word are assigned identical weights.

the similarity between $\mathcal{S}_1$ and $\mathcal{S}_2$ depends on the number of words from $\mathcal{S}_1$ that appear in $\mathcal{S}_2$, divided by the length of $\mathcal{S}_2$ (note that each word may carry a different weight). In the subsequent iterations, each word $\mathcal{W} \in \mathcal{S}_1$ contributes to the similarity of $\mathcal{S}_1$ to $\mathcal{S}_2$ a value between 0 and 1, indicating its affinity to $\mathcal{S}_2$, instead of voting either 0 (if $\mathcal{W} \in \mathcal{S}_2$) or 1 (if $\mathcal{W} \notin \mathcal{S}_2$). Analogously, sentences contribute values to word similarity.

One may view the iterations as successively capturing parameterized "genealogical" relationships. Let words that share contexts be called direct relatives; then words that share neighbors (have similar cooccurrence patterns) are *once-removed* relatives. These two family relationships are captured by the first iteration, and also by most traditional similarity measures, which are based on cooccurrences. The second iteration then brings together *twice-removed* relatives. The third iteration captures higher similarity relationships, and so on. Note that the level of relationship here is a gradually consolidated real-valued quantity, and is dictated by the amount and the quality of the evidence gleaned from the corpus; it is not an all-or-none "relatedness" tag, as in genealogy.

The following simple example demonstrates the difference between our similarity measure and pure cooccurrence-based similarity measures, which cannot capture higher-order relationships. Consider the set of three sentence fragments:

s1: *eat banana*

s2: *taste banana*

s3: *eat apple*

In this "corpus," the similarity of *taste* and *apple*, according to the cooccurrence-based methods, is 0, because the contexts of these two words are disjoint. In comparison, our iterative algorithm will capture some similarity:

- *Initialization.* Every word is similar to itself only.

- *First iteration.* The sentences *eat banana* and *eat apple* have similarity of 0.5, because of the common word *eat*. Furthermore, the sentences *eat banana* and *taste banana* have similarity 0.5:
  - *banana* is learned to be similar to *apple* because of their common usage (*eat banana* and *eat apple*);
  - *taste* is similar to *eat* because of their common usage (*taste banana* and *eat banana*);
  - *taste* and *apple* are not similar (yet).
- *Second iteration.* The sentence *taste banana* has now some similarity to *eat apple*, because in the previous iteration *taste* was similar to *eat* and *banana* was similar to *apple*. The word *taste* is now similar to *apple* because the *taste* sentence (*taste banana*) is similar to the *apple* sentence (*eat apple*). Yet, *banana* is more similar to *apple* than *taste*, because the similarity value of *banana* and *apple* further increases in the second iteration.

This simple example demonstrates the transitivity of our similarity measure, which allows it to extract high-order contextual relationships. In more complex situations, the transitivity-dependent spread of similarity is slower, because each word is represented by many more sentences.

The most important properties of the similarity computation algorithm are convergence (see appendix A.2), and utility in supporting disambiguation (described in section 2); three other properties are as follows. First, word similarity computed according to the above algorithm is asymmetric. For example, *drug* is more similar to *traffic* than *traffic* is to *drug*, because *traffic* is mentioned more frequently in *drug* contexts than *drug* is mentioned in contexts of *traffic* (which has many other usages). Likewise, sentence similarity is asymmetric: if $\mathcal{S}_1$ is fully contained in $\mathcal{S}_2$, then $\text{sim}(\mathcal{S}_1, \mathcal{S}_2) = 1$, whereas sim$(\mathcal{S}_2,\ \mathcal{S}_1) < 1$. Second, words with a small count in the training set will have unreliable similarity values. These, however, are multiplied by a very low weight when used in sentence similarity evaluation, because the frequency in the training set is taken into account in computing the word weights. Third, in the computation of sim$(\mathcal{W}_1,\ \mathcal{W}_2)$ for a very frequent $\mathcal{W}_2$, the set of its sentences is very large, potentially inflating the affinity of $\mathcal{W}_1$ to the sentences that contain $\mathcal{W}_2$. We counter this tendency by multiplying sim$(\mathcal{W}_1,\ \mathcal{W}_2)$ by a weight that is reciprocally related to the global frequency of $\mathcal{W}_2$.

1.3 Using similarity to tag the training set

Following convergence, each sentence in the training set is assigned the sense of its most similar sentence in one of the feedback sets of sense s_i, using the final sentence similarity matrix. Note that some sentences in the training set belong also to one of the feedback sets, because they contain words from the MRD definition of the target word. Those sentences are automatically assigned the sense of the feedback set to which they belong, since they are most similar to themselves. Note also that an original training-set sentence $\mathcal{S}$ can be attracted to a sentence F from a feedback set, even if S and F do not share any word, because of the transitivity of the similarity measure.

1.4 Learning the typical uses of each sense

We partition the examples of each sense into *typical use* sets, by grouping all the sentences that were attracted to the same feedback-set sentence. That sentence, and all the original sentences attracted to it, form a class of examples for a typical usage. Feedback-set examples that did not attract any original sentences are discarded. If the number of resulting classes is too high, further clustering can be carried out on the basis of the distance metric defined by $1 - \text{sim}(x,\ y)$, where sim$(x,\ y)$ are values taken from the final sentence similarity matrix.

A typical usage of a sense is represented by the affinity information generalized from its examples. For each word W, and each cluster C of examples of the same usage, we define:

$$\begin{aligned} \text{aff}(\mathcal{W},\ C) &= \max_{\mathcal{S}\in C} \text{aff}(\mathcal{W},\ \mathcal{S}) && (5)\\ &= \max_{\mathcal{S}\in C}\max_{\mathcal{W}_i\in\mathcal{S}} \text{sim}(\mathcal{W},\ \mathcal{W}_i) && (6) \end{aligned}$$

For each cluster we construct its affinity vector, whose i'th component indicates the affinity of word i to the cluster. It suffices to generalize the affinity information (rather than similarity), because new examples are judged on the basis of their similarity **to** each cluster: in the computation of sim$(\mathcal{S}_1,\ \mathcal{S}_2)$ (equation 3), the only information concerning $\mathcal{S}_2$ is its affinity values.

1.5 Testing new examples

Given a new sentence S containing a target word W, we determine its sense by computing the similarity of S to each of the previously obtained clusters C_k, and returning the sense of the most similar cluster:

$$\begin{aligned} \text{sim}(\mathcal{S}_{new},\ C_k) &= \sum_{\mathcal{W}\in\mathcal{S}_{new}} \text{weight}(\mathcal{W},\ \mathcal{S}_{new})\cdot\text{aff}(\mathcal{W},\ C_k) && (7)\\ \text{sim}(\mathcal{S}_{new},\ \text{s}_i) &= \max_{C\in\mathbf{s}_i} \text{sim}(\mathcal{S}_{new},\ C) && (8) \end{aligned}$$

2. Experimental evaluation of the method

We tested the algorithm on the Treebank-2 corpus, which contains 1 million words from the Wall Street Journal, 1989, and is considered a small corpus for the present task. As the MRD, we used a combination of the Webster, the Oxford and the WordNet online dictionaries (the latter used as a thesaurus only; see section 3.3). During the development and the tuning of the algorithm, we used the method of pseudo-words (Gale, Church,

Table 1
A summary of the experimental results on four polysemous words.

Word	Senses	Sample Size	Feedback Size	% correct per sense	% correct total
drug	narcotic	65	100	92.3	90.5
	medicine	83	65	89.1	
sentence	judgement	23	327	100	92.5
	grammar	4	42	50	
suit	court	212	1461	98.59	94.8
	garment	21	81	55	
player	performer	48	230	87.5	92.3
	participant	44	1552	97.7	

and Yarowsky, 1992; Schutze, 1992), to save the need for manual verification of the resulting sense tags.

The final algorithm was tested on a total of 500 examples of four polysemous words: *drug*, *sentence*, *suit*, and *player* (see Table 1). The relatively small number of polysemous words we studied was dictated by the size and nature of the corpus (we are currently testing additional words, using texts from the British National Corpus).

The average success rate of our algorithm was 92%. The original training set (before the addition of the feedback sets) consisted of a few dozen examples, in comparison to thousands of examples needed in other corpus-based methods (Schutze, 1992; Yarowsky, 1995).

We now present in detail several of the results obtained with the word *drug*. A plot of the improvement in the performance vs. iteration number appears in Figure 2. The success rate is plotted for each sense, and for the weighted average of both senses we considered (the weights are proportional to the number of examples of each sense).

Figure 3 shows how the similarity values develop with iteration number. For each example S of the *narcotic* sense of *drug*, the value of $\mathrm{sim}_n(\mathcal{S},\ narcotic)$ increases with n. Note that after several iterations the similarity values are close to 1, and, because they are bounded by 1, they cannot change significantly with further iterations.

Figure 4 compares the similarities of a *narcotic* example to the *narcotic* sense and to

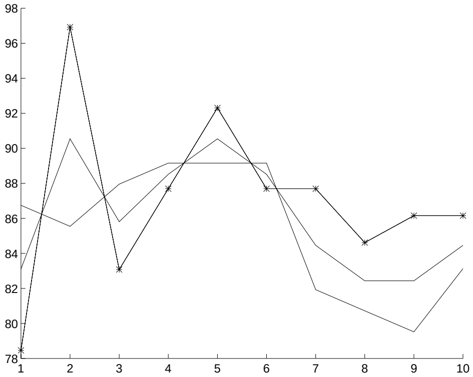

Figure 2
The *drug* experiment; the change in the disambiguation performance with iteration number is plotted separately for each sense. The asterisk marks the plot of the success rate for the *narcotic* sense.

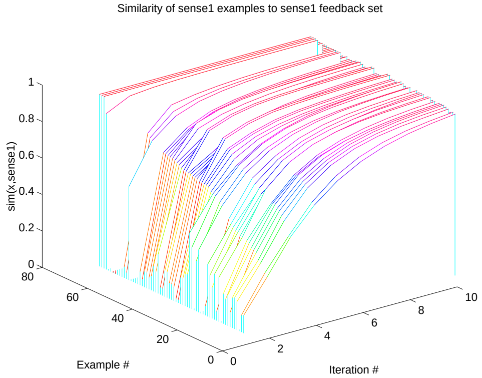

Figure 3
The *drug* experiment; example runs, sorted by the second-iteration similarity values.

the *medicine* sense, for each iteration. The *medicine* sense assignment, made in the first iteration, has been corrected in the following iterations.

Table 2 shows the most similar words found for the words with the highest weights in the *drug* example (low-similarity words have been omitted). Note that the similarity is contextual, and is affected by the polysemous target word. For example, *trafficking* was found to be similar to *crime*, because in *drug* contexts the expressions *drug trafficking* and *crime* are highly related. In general, *trafficking* and *crime* need not be similar, of course.

Table 2
The *drug* experiment; the nearest neighbors of the highest-weight words.

Word	Most contextually similar words
The *medicine* sense:	
medication	antibiotic blood prescription medicine percentage pressure
prescription	analyst antibiotic blood campaign introduction law line-up medication medicine percentage print profit publicity quarter sedative state television tranquilizer use
medicine	prescription campaign competition dollar earnings law manufacturing margin print product publicity quarter result sale saving sedative staff state television tranquilizer unit use
disease	antibiotic blood line-up medication medicine prescription
symptom	hypoglycemia insulin warning manufacturer product plant animal death diabetic evidence finding metabolism study
insulin	hypoglycemia manufacturer product symptom warning death diabetic finding report study
tranquilizer	campaign law medicine prescription print publicity sedative television use analyst profit state
dose	appeal death impact injury liability manufacturer miscarriage refusing ruling diethylstilbestrol hormone damage effect female prospect state
The *narcotic* sense:	
consumer	distributor effort cessation consumption country reduction requirement victory battle capacity cartel government mafia newspaper people
mafia	terrorism censorship dictatorship newspaper press brother nothing aspiration assassination editor leader politics rise action country doubt freedom mafioso medium menace solidarity structure trade world
terrorism	censorship doubt freedom mafia medium menace newspaper press solidarity structure
murder	capital-punishment symbolism trafficking furor killing substance crime restaurant law bill case problem
menace	terrorism freedom solidarity structure medium press censorship country doubt mafia newspaper way attack government magnitude people relation threat world
trafficking	crime capital-punishment furor killing murder restaurant substance symbolism
dictatorship	aspiration brother editor mafia nothing politics press assassination censorship leader newspaper rise terrorism
assassination	brother censorship dictatorship mafia nothing press terrorism aspiration editor leader newspaper politics rise
laundering	army lot money arsenal baron economy explosive government hand materiel military none opinion portion talk
censorship	mafia newspaper press terrorism country doubt freedom medium menace solidarity structure

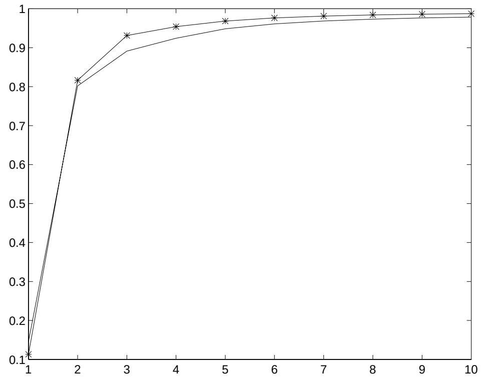

Figure 4
The *drug* experiment; the similarity between a *narcotic*-sense example to each of the two senses. The asterisk marks the plot for the *narcotic* sense. The sentence was *The American people and their government also woke up too late to the menace drugs posed to the moral structure of their country.*
The word *menace* which is a hint for the *narcotic* sense in this sentence, did not help in the first iteration, because it did not appear in the *narcotic* feedback set at all. Thus, in iteration 1, the similarity of this sentence to the *medicine* sense was 0.15, vs. similarity of 0.1 to the *narcotic* sense. In iteration 2, *menace* was learned to be similar to other *narcotic*-related words, yielding a small advantage for the *narcotic* sense. In iteration 3, further similarity values were updated, and there was a clear advantage to the *narcotic* sense (0.93, vs. 0.89 for *medicine*).

3. Discussion

We now discuss in some detail the choices made at the different stages of the development of the present method, and its relationship to some of the previous works on word sense disambiguation.

3.1 Flexible sense distinctions

The possibility of strict definition of each sense of a polysemous word, and the possibility of unambiguous assignment of a given sense in a given situation are, in themselves, nontrivial issues in philosophy (Quine, 1960) and linguistics (Weinreich, 1980; Cruse, 1986). Different dictionaries often disagree on the definitions; the split into senses may also depend on task at hand. Thus, it is important to maintain the possibility of flexible distinction of the different senses, e.g., by letting this distinction be determined by an external knowledge source such as a thesaurus or a dictionary. Although this requirement may seem trivial, most corpus-based methods do not, in fact, allow such flexibility. For example, defining the senses by the possible translations of the word (Dagan and Itai, 1991; Brown et al., 1991; Gale, Church, and Yarowsky, 1992), by the Roget's categories (Yarowsky, 1992), or by clustering (Schutze, 1992) yields a grouping that does not always conform to the desired sense distinctions. In comparison, our reliance on the MRD the definition of senses in the initialization of the learning process guarantees the required flexibility in setting the sense distinctions. Pure MRD-based methods allow the same flexibility, but do not yield good results, because the definitions alone do not contain enough information for disambiguation.

3.2 Sentence features

Different polysemous words may benefit from different types of features of the context sentences. Polysemous words for which distinct senses tend to appear in different topics can be disambiguated using single words as the context features, as we did here. Disambiguation of other polysemous words may require taking the sentence structure into account, using n-grams or syntactic constructs as features. This additional information can be incorporated into our method, by (1) extracting features such as nouns, verbs, adjectives of the target word, bi-grams, tri-grams, and subject-verb or verb-object pairs, (2) discarding features with a low weight (cf. section A.3), and (3) using the remaining features instead of single words (i.e., by representing a sentence by the set of significant features it contains, and a feature — by the set of sentences in which it appears).

3.3 Using WordNet

The initialization of the word similarity matrix using WordNet (a hand-crafted semantic network arranged in a hierarchical structure; (Miller et al., 1993)) may seem to be advantageous over simply setting it to the identity matrix, as we have done. To compare these two approaches, we tried to set the initial similarity between two words to the WordNet path length between their nodes (Lee, Kim, and Lee, 1993), and then learn the similarity values iteratively. This, however, led to worse performance than the simple identity-matrix initialization.

There are several possible reasons for the poor performance of WordNet in this comparison. First, WordNet is not designed to capture contextual similarity. For example, in WordNet, *hospital* and *doctor* have no common ancestor, and hence their similarity is 0, while *doctor* and *lawyer* are quite similar, because both designate professionals, humans, and living things. Note that, contextually, *doctor* should be more similar to *hospital* than to *lawyer*. Second, we found that the WordNet similarity values dominated the contextual similarity computed in the iterative process, preventing the transitive effects of contextual similarity from taking over. Third, the tree distance in itself does not always correspond to the intuitive notion of similarity, because different concepts appear at different level of abstraction, and have a different number of nested sub-concepts. For example, a certain distance between two nodes may result from: (1) the nodes being semantically close, but separated by a large distance, stemming from a high level of detail in the related synsets, or from (2) the nodes being semantically far from each other.[4]

3.4 Ignoring irrelevant examples

The feedback sets we use in training the system may contain noise, in the form of irrelevant examples that are collected along with the relevant and useful ones. For instance, in one of the definitions of *bank* in WordNet, we find *bar*, which, in turn, has many other senses that are not related to *bank*. Although these unrelated senses contribute examples to the feedback set, our system is hardly affected by this noise, because we do not collect statistics on the feedback sets (i.e., our method is not based on

[4] Resnik (1995) recently suggested to overcome this particular difficulty by a different measure that takes into account the informativeness of the most specific common ancestor of the two words.

mere cooccurrence frequencies, as most other corpus-based methods are). The relevant examples in the feedback set of the sense $\mathbf{s}_i$ will attract the examples of $\mathbf{s}_i$; the irrelevant examples, will not attract the examples of $\mathbf{s}_i$, but neither will they do damage, because they are not expected to attract examples of $\mathbf{s}_j$ $(j \neq i)$.

3.5 Related work

3.5.1 The knowledge acquisition bottleneck. Brown et al. (1991) and Gale et al. (1992) used the translations of the ambiguous word in a bilingual corpus as sense tags. This does not obviate the need for manual work, as producing bilingual corpora requires manual translation work. Dagan and Itai (1991) used a bilingual lexicon and a monolingual corpus, to save the need for translating the corpus. The problem remains, however, that the word translations do not necessarily overlap with the desired sense distinctions.

Schutze (1992) clustered the examples in the training set, and manually assigned each cluster a sense by observing 10-20 members of the cluster. Each sense was usually represented by several clusters. Although this approach significantly decreased the need for manual intervention, about a hundred examples had still to be tagged manually for each word. Moreover, the resulting clusters did not necessarily correspond to the desired sense distinctions.

Yarowsky (1992) learned discriminators for each Roget's category, saving the need to separate the training set into senses. However, using such hand-crafted categories usually leads to a coverage problem for specific domains, or for domains other than the one for which the list of categories has been prepared.

Using MRDs for WSD was suggested in (Lesk, 1986); several researchers subsequently continued and improved this line of work (Krovetz and Croft, 1989; Guthrie et al., 1991; Veronis and Ide, 1990). Unlike the information in a corpus, the information in the MRD definitions is presorted into senses. However, as noted above, the MRD definitions alone do not contain enough information to allow reliable disambiguation. Recently, Yarowsky (1995) combined a MRD and a corpus in a bootstrapping process. In that work, the definition words were used as initial sense indicators, tagging automatically the target word examples containing them. These tagged examples were then used as seed examples in the bootstrapping process. In comparison, we suggest to combine further the corpus and the MRD by use *all* the corpus examples of the MRD definition words, instead of those words alone. This yields much more sense-presorted training information.

3.5.2 The problem of sparse data. Most previous works define word similarity based on cooccurrence information, and hence face a severe problem of sparse data. Many of the possible cooccurrences are not observed even in a very large corpus (Church and Mercer, 1993). Our algorithm addresses this problem in two ways. First, we replace the all-or-none indicator of cooccurrence by a graded measure of contextual similarity. Our measure of similarity is transitive, allowing two words to be considered similar even if they are neither observed in the same sentence, nor share neighbor words. Second, we extend the training set by adding examples of related words. The performance of our system compares favorably to that of systems trained on sets larger by a factor of 100 (the results described in section 2 were obtained following learning from several dozen examples, in comparison to thousands of examples in other automatic methods).

Traditionally, the problem of sparse data is approached by estimating the probability of unobserved cooccurrences using the actual cooccurrences in the training set. This can be done by smoothing the observed frequencies (Church and Mercer, 1993), or by class-based methods (Brown et al., 1991; Pereira and Tishby, 1992; Pereira, Tishby, and Lee, 1993; Hirschman, 1986; Resnik, July 1992; Brill et al., June 1990; Dagan, Marcus, and Markovitch, 1993). In comparison to these approaches, we use similarity information throughout training, and not merely for estimating cooccurrence statistics. This allows the system to learn successfully from very sparse data.

3.6 Summary

We have described an approach to WSD that combines a corpus and a MRD to generate an extensive data set for learning similarity-based disambiguation. Our system combines the advantages of corpus-based approaches (large number of examples) with those of the MRD-based approaches (data pre-sorted by senses), by using the MRD definitions to direct the extraction of training information (in the form of feedback sets) from the corpus.

In our system, a word is represented by the set of sentences in which it appears. Accordingly, words are considered similar if they appear in similar sentences, and sentences are considered similar if they contain similar words. Applying this definition iteratively yields a transitive measure of similarity under which two sentences may be considered similar even if they do not share any word, and two words may be considered as similar even if they do not share neighbor words. Our experiments show that the resulting alternative to raw cooccurrence-based similarity leads to better performance on very sparse data.

A. Appendix

A.1 Stopping conditions of the iterative algorithm

Let f_i be the increase in the similarity value in iteration i:

$$f_i(\mathcal{X},\mathcal{Y}) = \mathrm{sim}_i(\mathcal{X},\ \mathcal{Y}) - \mathrm{sim}_{i-1}(\mathcal{X},\ \mathcal{Y}) \tag{9}$$

where X, Y can be either words or sentences. For each item X, the algorithm stops updating its similarity values to other items (that is, updating its row in the similarity matrix) in the first iteration that satisfies $max_{\mathcal{Y}} f_i(\mathcal{X}, \mathcal{Y}) \leq \epsilon$, where $\epsilon > 0$ is a preset threshold.

According to this stopping condition, the algorithm terminates after at most $\frac{1}{\epsilon}$ iterations (otherwise, in $\frac{1}{\epsilon}$ iterations with each $f_i > \epsilon$, we obtain sim$(\mathcal{X},\ \mathcal{Y}) > \epsilon \cdot \frac{1}{\epsilon} = 1$, in contradiction to upper bound of 1 on the similarity values; see section A.2).

We found that the best results are obtained within three iterations. After that, the disambiguation results tend not to change significantly, although the similarity values may continue to increase. Intuitively, the transitive exploration of similarities is exhausted after three iterations.

A.2 Proofs

In the following, X, Y can be either words or sentences.

Theorem 1

Similarity is bounded: sim$_n(\mathcal{X},\ \mathcal{Y}) \leq 1$

Proof

By induction on the number of iteration. At the first iteration, sim$_0(\mathcal{X},\ \mathcal{Y}) \leq 1$, by initialization. Assume that the claim holds for n, and prove for $n+1$:

$$\begin{aligned} sim_{n+1}(x, y) &= \sum_{x_j \in x} weight(x_j, x) \max_{y_k \in y} sim_n(x_j, y_k) \\ &\leq \sum_{x_j \in x} weight(x_j, x) \cdot 1 \quad \text{(by the induction hypothesis)} \\ &= 1 \end{aligned}$$

■

Theorem 2

Similarity is reflexive: $\forall \mathcal{X},\ \ \text{sim}(\mathcal{X},\ \mathcal{X}) = 1$

Proof

By induction on the number of iteration. $\mathrm{sim}_0(\mathcal{X},\ \mathcal{X}) = 1$, by initialization. Assume that the claim holds for n, and prove for $n+1$:

$$\begin{aligned}
\mathrm{sim}_{n+1}(\mathcal{X},\ \mathcal{X}) &= \sum_{\mathcal{X}_i \in \mathcal{X}} \mathrm{weight}(\mathcal{X}_i,\ \mathcal{X}) \cdot \max_{\mathcal{X}_j \in \mathcal{X}} \mathrm{sim}_n(\mathcal{X}_i,\ \mathcal{X}_j) \\
&\geq \sum_{\mathcal{X}_i \in \S} \mathrm{weight}(\mathcal{X}_i,\ \mathcal{X}) \cdot \mathrm{sim}_n(\mathcal{X}_i,\ \mathcal{X}_i) \\
&= \sum_{x_i \in x} \mathrm{weight}(\mathcal{X}_i,\ \mathcal{X}) \cdot 1 \quad \text{(by the induction hypothesis)} \\
&= 1
\end{aligned}$$

Thus, $\mathrm{sim}_{n+1}(\mathcal{X},\ \mathcal{X}) \geq 1$. By theorem 1, $\mathrm{sim}_{n+1}(\mathcal{X},\ \mathcal{X}) \leq 1$, so $\mathrm{sim}_{n+1}(\mathcal{X},\ \mathcal{X}) = 1$. ■

Theorem 3

Similarity $\mathrm{sim}_n(\mathcal{X},\ \mathcal{Y})$ is a non-decreasing function of the number of iteration n.

Proof

By induction on the number of iteration. Consider the case of $n = 1$: $\mathrm{sim}_1(\mathcal{X},\ \mathcal{Y}) \geq \mathrm{sim}_0(\mathcal{X},\ \mathcal{Y})$ (if $\mathrm{sim}_0(\mathcal{X},\ \mathcal{Y}) = 1$, then $\mathcal{X} = \mathcal{Y}$, and $\mathrm{sim}_1(\mathcal{X},\ \mathcal{Y}) = 1$ as well; else $\mathrm{sim}_0(\mathcal{X},\ \mathcal{Y}) = 0$ and $\mathrm{sim}_1(\mathcal{X},\ \mathcal{Y}) \geq 0 = \mathrm{sim}_0(\mathcal{X},\ \mathcal{Y})$). Now, assume that the claim holds for n, and prove for $n+1$:

$$\begin{aligned}
\mathrm{sim}_{n+1}(\mathcal{X},\ \mathcal{Y}) \quad &- \quad \mathrm{sim}_n(\mathcal{X},\ \mathcal{Y}) = \\
&= \sum_{\mathcal{X}_j \in \mathcal{X}} \mathrm{weight}(\mathcal{X}_j,\ \mathcal{X}) \cdot \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_n(\mathcal{X}_j,\ \mathcal{Y}_k) - \sum_{\mathcal{X}_j \in \mathcal{X}} \mathrm{weight}(\mathcal{X}_j,\ \mathcal{X}) \cdot \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_{n-1}(\mathcal{X}_j,\ \mathcal{Y}_k) \\
&\geq \sum_{\mathcal{X}_j \in \mathcal{X}} \mathrm{weight}(\mathcal{X}_j,\ \mathcal{X}) \cdot \left(\max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_n(\mathcal{X}_j,\ \mathcal{Y}_k) - \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_{n-1}(\mathcal{X}_j,\ \mathcal{Y}_k) \right) \\
&\geq 0
\end{aligned}$$

The last inequality hold because, by the induction hypothesis,

$$\forall \mathcal{X}_j, \mathcal{Y}_k,\ \ \mathrm{sim}_n(\mathcal{X}_j,\ \mathcal{Y}_k) \quad \geq \quad \mathrm{sim}_{n-1}(\mathcal{X}_j,\ \mathcal{Y}_k)$$

$$\begin{aligned}\max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_n(\mathcal{X}_j, \mathcal{Y}_k) &\geq \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_{n-1}(\mathcal{X}_j, \mathcal{Y}_k) \\ \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_n(\mathcal{X}_j, \mathcal{Y}_k) - \max_{\mathcal{Y}_k \in \mathcal{Y}} \mathrm{sim}_{n-1}(\mathcal{X}_j, \mathcal{Y}_k) &\geq 0\end{aligned}$$

Thus, all the items under the sum are nonnegative, and so must be their weighted average. As a consequence, we may conclude that the iterative estimation of similarity converges. ■

A.3 Word weights

In our algorithm, the weight of a word estimates its expected contribution to the disambiguation task, and the extent to which the word is indicative in sentence similarity. The weights do not change with iterations. They are used to reduce the number of features to a manageable size, and to exclude words that are expected to be given unreliable similarity values. The weight of a word is a product of several factors: frequency in the corpus, the bias inherent in the training set, distance from the target word, and part of speech label:

1. *Global frequency*. Frequent words are less informative of the sense and of the sentence similarity (e.g., the appearance of *this* in two different sentences does not indicate similarity between them, and does not indicate the sense of any target word). The contribution of frequency is $\max\{0, 1 - \frac{\mathrm{freq}(w)}{\mathrm{max5}_{\mathcal{X}}\mathrm{freq}(\mathcal{X})}$, where $\mathrm{max5}_{\mathcal{X}}\mathrm{freq}(\mathcal{X})$ is a function of the five highest frequencies in the corpus. This factor excludes only the most frequent words from further consideration. As long as the frequencies are not very high, it does not label $\mathcal{W}_1$ whose frequency is twice that of $\mathcal{W}_2$ as less informative.

2. *Log likelihood factor*. Words that are indicative of the sense usually appear in the training set more than what would have been expected from their

frequency in the general corpus. The log likelihood factor captures this tendency. It is computed as

$$\log \frac{\Pr(\mathcal{W}_i \mid \mathcal{W})}{\Pr(\mathcal{W}_i)} \tag{10}$$

where $\Pr(\mathcal{W}_i)$ is estimated from the frequency of W in the entire corpus, and $\Pr(\mathcal{W}_i \mid \mathcal{W})$ — from the frequency of $\mathcal{W}_i$ in the training set, given the examples of the current ambiguous word $\mathcal{W}$ (cf. (Gale, Church, and Yarowsky, 1992)).[5] To avoid poor estimation for words with a low count in the training set, we multiply the log likelihood by $\min\{1, \frac{count(\mathcal{W})}{10}\}$ where *count*(W) is the number of occurrences of W in the training set.

3. *Part of speech.* Each part of speech is assigned an initial weight (1.0 for nouns and 0.6 for verbs).

4. *Distance from the target word.* Context words that are far from the target word are less indicative than nearby ones. The contribution of this factor is reciprocally related to the normalized distance.

The total weight of a word is the product of the above factors, each normalized by the sum of factors of the words in the sentence: $\text{weight}(\mathcal{W}_i,\ \mathcal{S}) = \frac{\text{factor}(\mathcal{W}_i,\ \mathcal{S})}{\sum_{\mathcal{W}_j \in \mathcal{S}} \text{factor}(\mathcal{W}_j,\ \mathcal{S})}$, where $\text{factor}(.,\ .)$ is the weight before normalization.

A.4 Other uses of context similarity

The similarity measure developed in the present paper can be used for tasks other than word sense disambiguation. Here, we illustrate a possible application to automatic construction of a thesaurus.

[5] Because this estimate is unreliable for words with low frequencies in each sense set, Gale et al. (1992) suggested to interpolate between probabilities computed within the sub-corpus and probabilities computed over the entire corpus. In our case, the denominator is the frequency in the general corpus instead of the frequency in the sense examples, so it is more reliable.

Following the training phase for a word X, we have a word similarity matrix for the words in the contexts of X. Using this matrix, we construct for each sense s_i of X a set of related words, R:

1. Initialize R to the set of words appearing in the MRD definition of s_i;
2. Extend R recursively: for each word in R added in the previous step, add its k nearest neighbors, using the similarity matrix.
3. Stop when no new words (or too few new words) are added.

Upon termination, output for each sense s_i the set of its contextually similar words R.

Acknowledgments
We thank Dan Roth for comments on a draft of this paper.

References

Brill, E., D. Magerman, M. Marcus, and B. Santorini. June 1990. Deducing linguistic structure from the statistics of large corpora. *DARPA speech and natural language workshop*.

Brown, P., S. D. Pietra, V. D. Pietra, and R. L. Mercer. 1991. Word sense disambiguation using statistical methods. In *Proceedings of the 29th Annual Meeting of the ACL*, pages 264–270.

Church, K. W. and R. L. Mercer. 1993. Introduction to the special issue in computational linguistics using large corpora. *Computational Linguistics*, 19:1–24.

Cruse, D. A. 1986. *Lexical Semantics*. Cambridge University Press, Cambridge, England.

Dagan, I. and A. Itai. 1991. Two languages are more informative than one. In *Proceedings of the 29th Annual Meeting of the ACL*, pages 130–137.

Dagan, I., S. Marcus, and S. Markovitch. 1993. Contextual word similarity and estimation from sparse data. In *Proceedings of the 31st Annual Meeting of the ACL*, pages 164–174.

Gale, W., K. Church, and D. Yarowsky. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

Guthrie, J. A., L. Guthrie, Y. Wilks, and H. Aidinejad. 1991. Subject-dependent cooccurrence and word sense disambiguation. In *Proceedings of the 29th Annual Meeting of the ACL*, pages 146–152.

Hirschman, L. 1986. Discovering sublanguage structure. In R. Grishman and R. Kittredge, editors, *Analyzing Language in Restricted Domains: Sublanguage description and processing*. Lawrence Erlbaum, Hillsdale, NJ, pages 211–234.

Krovetz, R. and W. B. Croft. 1989. Word sense disambiguation using machine readable dictionaries. In *Proceedings of ACM SIGIR'89*, pages 127–136.

Lee, J. H., M. H. Kim, and Y. J. Lee. 1993. Information retrieval based on conceptual distance in IS-A hierarchies. *Journal of Documentation*, 49:188–207.

Lesk, M. 1986. Automatic sense disambiguation: How to tell a pine cone from an ice cream cone. In *Proceedings of the 1986 ACM SIGDOC Conference*, pages 24–26.

Miller, G. A., R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. 1993. Introduction to WordNet: an on-line lexical database. CSL 43, Cognitive Science Laboratory, Princeton University, Princeton, NJ.

Pereira, F. and N. Tishby. 1992. Distibutional similarity, phase transitions and hierarchical clustering. In *Working Notes of the AAAI Fall Symposium on probabilistic approaches to natural language*, pages 108–112.

Pereira, F., N. Tishby, and L. Lee. 1993. Distibutional clustering of English words.

In *Proceedings of the 31st Annual Meeting of the ACL*, pages 183–190.

Quine, W. V. O. 1960. *Word and Object*. MIT Press, Cambridge, MA.

Resnik, P. July 1992. WordNet and distribuitional analysis: A class-based approach to lexical discovery. In *AAAI workshop on statistically-based natural language processing techniques*, pages 56–64.

Resnik, P. June 1995. Disambiguating noun groupings with respect to wordnet senses. In *Third workshop on very large corpora*.

Schutze, H. 1992. Dimensions of meaning. In *Proceedings of Supercomputing Symposium*, pages 787–796, Minneapolis, MN.

Veronis, J. and N. Ide. 1990. Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *Proceedings of COLING-90*, pages 389–394.

Weinreich, U. 1980. *On Semantics*. University of Pennsylvania Press, Philadelphia, PA.

Yarowsky, D. 1992. Word sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of COLING-92*, pages 454–460, Nantes.

Yarowsky, D. 1994. Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of the ACL*, Las Cruces, NM.

Yarowsky, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the ACL*, pages 189–196, Cambridge, MA.