Discovering Corpus-Specific Word Senses

Appeared in EACL 2003, Budapest, Hungary
Conference Companion (research notes and demos), pp. 79-82

Beate Dorow
Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart, Germany
beate.dorow@ims.uni-stuttgart.de

Dominic Widdows
Center for the Study of Language and Information
Stanford University, California
dwiddows@csli.stanford.edu

Abstract

This paper presents an unsupervised algorithm which automatically discovers word senses from text. The algorithm is based on a graph model representing words and relationships between them. Sense clusters are iteratively computed by clustering the local graph of similar words around an ambiguous word. Discrimination against previously extracted sense clusters enables us to discover new senses. We use the same data for both recognising and resolving ambiguity.

1 Introduction

This paper describes an algorithm which automatically discovers word senses from free text and maps them to the appropriate entries of existing dictionaries or taxonomies.

Automatic word sense discovery has applications of many kinds. It can greatly facilitate a lexicographer's work and can be used to automatically construct corpus-based taxonomies or to tune existing ones. The same corpus evidence which supports a clustering of an ambiguous word into distinct senses can be used to decide which sense is referred to in a given context (Schütze, 1998).

This paper is organised as follows. In section 2, we present the graph model from which we discover word senses. Section 3 describes the way we divide graphs surrounding ambiguous words into different areas corresponding to different senses, using Markov clustering (van Dongen, 2000). The quality of the Markov clustering depends strongly on several parameters such as a granularity factor and the size of the local graph. In section 4, we outline a word sense discovery algorithm which bypasses the problem of parameter tuning. We conducted a pilot experiment to examine the performance of our algorithm on a set of words with varying degree of ambiguity. Section 5 describes the experiment and presents a sample of the results. Finally, section 6 sketches applications of the algorithm and discusses future work.

2 Building a Graph of Similar Words

The model from which we discover distinct word senses is built automatically from the British National corpus, which is tagged for parts of speech. Based on the intuition that nouns which co-occur in a list are often semantically related, we extract contexts of the form *Noun, Noun,... and/or Noun*, e.g. "genomic DNA from *rat*, *mouse* and *dog*".

Following the method in (Widdows and Dorow, 2002), we build a graph in which each node represents a noun and two nodes have an edge between them if they co-occur in lists more than a given number of times [1].

Following Lin's work (1998), we are currently investigating a graph with verb-object, verb-subject and modifier-noun-collocations from which it is possible to infer more about the senses of systematically polysemous words. The word sense clustering algorithm as outlined below can be applied to any kind of similarity measure based on any set of features.

[1] Simple cutoff functions proved unsatisfactory because of the bias they give to more frequent words. Instead we link each word to its top n neighbors where n can be determined by the user (cf. section 4).

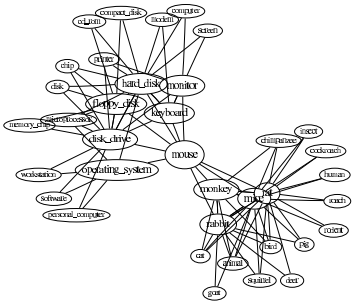

Figure 1: Local graph of the word *mouse*

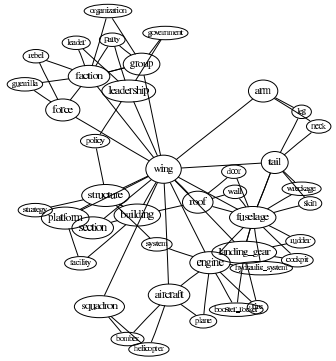

Figure 2: Local graph of the word *wing*

3 Markov Clustering

Ambiguous words link otherwise unrelated areas of meaning. E.g. *rat* and *printer* are very different in meaning, but they are both closely related to different meanings of *mouse*. However, if we remove the *mouse*-node from its local graph illustrated in figure 1, the graph decomposes into two parts, one representing the electronic device meaning of *mouse* and the other one representing its animal sense. There are, of course, many more types of polysemy (cf. e.g. (Kilgarriff, 1992)). As can be seen in figure 2, *wing* "part of a bird" is closely related to *tail*, as is *wing* "part of a plane". Therefore, even after removal of the *wing*-node, the two areas of meaning are still linked via *tail*. The same happens with *wing* "part of a building" and *wing* "political group" which are linked via *policy*. However, whereas there are many edges *within* an area of meaning, there is only a small number of (weak) links *between* different areas of meaning. To detect the different areas of meaning in our local graphs, we use a cluster algorithm for graphs (Markov clustering, MCL) developed by van Dongen (2000). The idea underlying the MCL-algorithm is that random walks within the graph will tend to stay in the same cluster rather than jump between clusters.

The following notation and description of the MCL algorithm borrows heavily from van Dongen (2000). Let G_w denote the local graph around the ambiguous word w. The adjacency matrix M_{G_w} of a graph G_w is defined by setting $(M_{G_w})_{pq}$ equal to the weight of the edge between nodes v_p and v_q. Normalizing the columns of M_{G_w} results in the Markov Matrix T_{G_w} whose entries $(T_{G_w})_{pq}$ can be interpreted as transition probability from v_q to v_p. It can easily be shown that the k-th power of T_{G_w} lists the probabilities $(T^k_{G_w})_{pq}$ of a path of length k starting at node v_q and ending at node v_p.

The MCL-algorithm simulates flow in G_w by iteratively recomputing the set of transition probabilities via two steps, expansion and inflation. The expansion step corresponds with taking the k-th power of T_{G_w} as outlined above and allows nodes to see new neighbours. The inflation step takes each matrix entry to the r-th power and then rescales each column so that the entries sum to 1.Via inflation, popular neighbours are further supported at the expense of less popular ones.

Flow within dense regions in the graph is concentrated by both expansion and inflation. Eventually, flow between dense regions will disappear, the matrix of transition probabilities T_{G_w} will converge and the limiting matrix can be interpreted as a clustering of the graph.

4 Word Sense Clustering Algorithm

The output of the MCL-algorithm strongly depends on the inflation and expansion parameters r and k as well as the size of the local graph which serves as input to MCL.

An appropriate choice of the inflation param-

eter r can depend on the ambiguous word w to be clustered. In case of homonymy, a small inflation parameter r would be appropriate. However, there are ambiguous words with more closely related senses which are metaphorical or metonymic variations of one another. In that case, the different regions of meaning are more strongly interlinked and a small power coefficient r would lump different meanings together.

Usually, one sense of an ambiguous word w is much more frequent than its other senses present in the corpus. If the local graph handed over to the MCL process is small, we might miss some of w's meanings in the corpus. On the other hand, if the local graph is too big, we will get a lot of noise.

Below, we outline an algorithm which circumvents the problem of choosing the right parameters. In contrast to pure Markov clustering, we don't try to find a complete clustering of G_w into senses at once. Instead, in each step of the iterative process, we try to find the most disctinctive cluster c of G_w (i.e. the most distinctive meaning of w) only. We then recompute the local graph G_w by discriminating against c's features. This is achieved, in a manner similar to Pantel and Lin's (2002) sense clustering approach, by removing c's features from the set of features used for finding similar words. The process is stopped if the similarity between w and its best neighbour under the reduced set of features is below a fixed threshold.

Let F be the set of w's features, and let L be the output of the algorithm, i.e. a list of sense clusters initially empty. The algorithm consists of the following steps:

1. Compute a small local graph G_w around w using the set of features F. If the similarity between w and its closest neighbour is below a fixed threshold go to 6.
2. Recursively remove all nodes of degree one. Then remove the node corresponding with w from G_w.
3. Apply MCL to G_w with a fairly big inflation parameter r which is fixed.
4. Take the "best" cluster (the one that is most strongly connected to w in G_w before removal of w), add it to the final list of clusters L and remove/devalue its features from F.
5. Go back to 1 with the reduced/devalued set of features F.
6. Go through the final list of clusters L and assign a name to each cluster using a broad-coverage taxonomy (see below). Merge semantically close clusters using a taxonomy-based semantic distance measure (Budanitsky and Hirst, 2001) and assign a class-label to the newly formed cluster.
7. Output the list of class-labels which best represent the different senses of w in the corpus.

The local graph in step 1 consists of w, the n_1 neighbours of w and the n_2 neighbours of the neighbours of w. Since in each iteration we only attempt to find the "best" cluster, it suffices to build a relatively small graph in 1. Step 2 removes noisy strings of nodes pointing away from G_w. The removal of w from G_w might already separate the different areas of meaning, but will at least significantly loosen the ties between them.

In our simple model based on noun co-occurrences in lists, step 5 corresponds to rebuilding the graph under the restriction that the nodes in the new graph not co-occur (or at least not very often) with any of the cluster members already extracted.

The class-labelling (step 6) is accomplished using the taxonomic structure of WordNet, using a robust algorithm developed specially for this purpose. The hypernym which subsumes as many cluster members as possible and does so as closely as possible in the taxonomic tree is chosen as class-label. The family of such algorithms is described in (Widdows, 2003).

5 Experimental Results

In this section, we describe an initial evaluation experiment and present the results. We will soon carry out and report on a more thorough analysis of our algorithm.

We used the simple graph model based on co-occurrences of nouns in lists (cf. section 2) for our experiment. We gathered a list of nouns with varying degree of ambiguity, from homonymy (e.g. *arms*) to systematic polysemy (e.g. *cherry*). Our algorithm was applied to each word in the list (with parameters $n_1 = 20, n_2 = 10, r = 2.0, k = 2.0$) in order to extract the top two sense clusters

only. We then determined the WordNet synsets which most adequately characterized the sense clusters. An extract of the results is listed in table 1.

Word	Sense clusters	Class-label
arms	knees trousers feet biceps hips elbows backs wings breasts shoulders thighs bones buttocks ankles legs inches wrists shoes necks	body part
	horses muskets charges weapons methods firearms knives explosives bombs bases mines projectiles drugs missiles uniforms	weapon
jersey	israel colombo guernsey luxembourg denmark malta greece belgium sweden turkey gibraltar portugal ireland mauritius britain cyprus netherlands norway australia italy japan canada kingdom spain austria zealand england france germany switzerland finland poland america usa iceland holland scotland uk	European country
	crucifix bow apron sweater tie anorak hose bracelet helmet waistcoat jacket pullover equipment cap collar suit fleece tunic shirt scarf belt	garment
head	voice torso back chest face abdomen side belly groin spine breast bill rump midhair hat collar waist tail stomach skin throat neck speculum	body part
	ceo treasurer justice chancellor principal founder president commander deputy administrator constable librarian secretary governor captain premier executive chief curator assistant committee patron ruler	person
oil	heat coal power water gas food wood fuel steam tax heating kerosene fire petroleum dust sand light steel telephone timber supply drainage diesel electricity acid air insurance petrol	object
	tempera gouache watercolour poster pastel collage acrylic	paint
lemon	bread cheese mint butter jam cream pudding yogurt sprinkling honey jelly toast ham chocolate pie syrup milk meat beef cake yoghurt grain	foodstuff
	hazel elder holly family virgin hawthorn	shrub
cherry	cedar larch mahogany water sycamore lime teak ash hornbeam oak walnut hazel pine beech alder thorn poplar birch chestnut blackthorn spruce holly yew laurel maple elm fir hawthorn willow	wood
	bacon cream honey pie grape blackcurrant cake banana	foodstuff

Table 1: Output of word sense clustering.

6 Applications and future research

The benefits of automatic, data-driven word sense discovery for natural language processing and lexicography would be very great. Here we only mention a few direct results of our work.

Our algorithm does not only recognise ambiguity, but can also be used to resolve it, because the features shared by the members of each sense cluster provide strong indication of which reading of an ambiguous word is appropriate given a certain context. This gives rise to an automatic, unsupervised word sense disambiguation algorithm which is trained on the data to be disambiguated.

The ability to map senses into a taxonomy using the class-labelling algorithm can be used to ensure that the sense-distinctions discovered correspond to recognised differences in meaning. This approach to disambiguation combines the benefits of both Yarowsky's (1995) and Schütze's (1998) approaches. Preliminary observations show that the different neighbours in Table 1 can be used to indicate with great accuracy which of the senses is being used.

Off-the-shelf lexical resources are rarely adequate for NLP tasks without being adapted. They often contain many rare senses, but not the same ones that are relevant for specific domains or corpora. The problem can be addressed by using word sense clustering to attune an existing resource to accurately describe the meanings used in a particular corpus.

We prepare an evaluation of our algorithm as applied to the collocation relationships (cf. section 2), and we plan to evaluate the uses of our clustering algorithm for unsupervised disambiguation more thoroughly.

References

A. Budanitsky. G. Hirst. 2001. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, June.

S. van Dongen. 2000. A cluster algorithm for graphs. Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science, Amsterdam, The Netherlands, May.

A. Kilgarriff. 1992. *Polysemy*. Ph.D. Thesis, University of Sussex, December.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL98*, Montreal, Canada, August.

P. Pantel. D. Lin. 2002. Discovering word senses from text. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, May.

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

D. Widdows. B. Dorow. 2002. A graph model for unsupervised lexical acquisition. In *COLING*, Taiwan, August.

D. Widdows. 2003. Unsupervised methods for developing taxonomies using syntactic and statistical information. In *HLT-NAACL (to appear)*, Edmonton, Canada.

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