Knowledge-Lean Approaches to Word Sense Disambiguation

Michael Gabilondo
CAP 6640: Natural Language Understanding

January 22, 2009

Abstract

This paper summarizes and discusses five different papers on knowledge-lean word sense disambiguation. Four of the papers have to do with Word Sense Induction (WSI), which is word sense disambiguation except with the additional task of discovering word senses automatically without relying on an external lexicon. The other paper is an example of word sense disambiguation that uses very little knowledge.

1 Introduction

Word Sense Disambiguation (WSD) is the task of selecting the correct sense of a polysemous word, usually with respect to a predefined set of senses from a dictionary. Word Sense Discrimination or Word Sense Induction (WSI) is a *knowledge lean* approach to disambiguation. These approaches differ from traditional word sense disambiguation in that they determine the set of senses of a polysemous word automatically from the unannotated training corpus. Therefore, they do not rely on an external lexicon and they also typically do not rely on other external sources of manually crafted knowledge. Potentially, one advantage of this is that it may discover very domain specific usages of words that may not be listed in available lexicons. The following sections summarize and discuss (Schütze, 1998), (Purandare and Pedersen, 2004), (Bordag, 2006) and (Pedersen and Bruce, 1998), which are different attempts at word sense discrimination or WSI, as well as (Yarowsky, 1995) which is WSD but relies on very little knowledge.

2 Schütze (1998)

Schütze divides the task of word sense disambiguation into two subtasks: word sense discrimination and sense labeling. In word sense discrimina-

tion, different usages of some polysemous word w are divided into a small number of clusters that each represent a sense of w. Sense labeling is the task of mapping senses from an outside knowledge source, such as WordNet (Fellbaum, 1998), to the different clusters. Because they carry out word sense discrimination without reference to any external lexicon, these sense clusters may not map perfectly to any particular set of predefined senses. Therefore, their paper only concentrates on word sense discrimination and not the automation of sense labeling: clusters are labeled manually for evaluation. They concluded that their method, in most cases, it performs above the baseline of labeling each word with the most common sense. They also describe a possible application in Information Retrieval (IR) of computing document-query similarity, for which reference to external senses is not needed.

Their algorithm is for word sense discrimination is called *context-group discrimination*. It is completely unsupervised in that the training corpus is unannotated and no lexicon is needed. The clusters created during context-group discrimination are called *context-groups* and the categorization of a usage of w in context c into one of the context-groups depends on the *context vector* of c, which is the centroid of the *word vectors* of the words in context c. Word vectors are high dimensional vectors in which the different dimensions represent cooccurrence frequencies with other words in the corpus; that is, given word vector $w = (f_{w_1}, f_{w_2}, ..., f_{w_n})$, f_{w_i}, for $1 \leq i \leq n$, represents the frequency with which w cooccurred with w_i. The centroid vectors of the context-groups, called *sense vectors*, represent different senses of w. To disambiguate words from a test corpus, a context vector is created for the word and it is assigned assigned to the cluster whose sense vector the context vector is closest to.

A context vector v_c is a representation of the local context c of some word w. It represents the *features* of the local context; in their paper, the features are the word vectors of the words around w. It is useful to make the distinction between *second order* context vectors and *first order* context vectors. The description of the context vectors above are second order. If the features were simply the words around w, rather than the word vectors of the words around w, then the context vector would be first order. A word vector of w_i can be thought of as the sum of the first order context vectors of w_i. The second order context vectors are therefore the sum of the sums of first order context vectors of the words in the local context. Second order context vectors have the effect of enriching the information about the local context by taking into account additional knowledge of where the words around w tend to be used.

The set of context vectors $\{v_{c_1}, v_{c_2}, ..., v_{c_n}\}$ for some word w appearing in contexts $c_1, c_2, ..., c_n$ is divided into a small number of predetermined clusters. This is done by using *group-average agglomerative clustering* (GAAC) and *Expectation Maximization* (EM). EM is locally optimal in that it guarantees that the centroid vector of each cluster is an optimal representative of the context vectors in that cluster. However, a bad choice of starting vectors for the algorithm will lead to a set of clusters that is not globally optimal. Therefore, GAAC is used to find a good starting point.

In agglomerative clustering, each element is initially in its own cluster. Clusters are then iteratively merged according to a goodness criterion function until some predetermined number of clusters has been reached. GAAC uses a criterion that is a combination of *single-link* and *complete-link* clustering. At each iteration, single-link clustering merges clusters C_1 and C_2 if there exists elements $e_1 \in C_1$ and $e_2 \in C_2$ such that for any other two elements $e_i \in C_m$ and $e_j \in C_n$, $distance(e_1, e_2) \leq distance(e_i, e_j)$. Complete-link clustering merges two clusters C_1 and C_2 whose merger $C_1 \cup C_2$ results in the smallest possible diameter if any possible merger. Single-link clustering tends to result in elongated clusters and complete-link clustering is strongly affected by outlying elements, as well as having a high time complexity. GAAC is a hybrid of these two methods.

The choice of clustering depends on the representation of the context vectors. Context vectors are thousands of dimensions, since their dimensions represent cooccurrence frequencies with other words in the corpus. And although word vectors can be sparse, context vectors are not because they are the sum of word vectors. Therefore, in addition to using the representation of context vectors as described previously, they also use *Singular Value Decomposition* (SVD) to reduce the dimensionality of the vector space where the context vectors exist, which is called *Word Space*; in this case, the dimensionality of the vectors in Word Space were reduced to 100. This dimensionality reduction has the effect of projecting nearby vectors closer to each other and far away vectors further away from each other. Therefore, in addition to the convenience of working with fewer dimensions, using *SVD-reduced* context vectors can potentially lead to more accurate clustering than word cooccurrence context vectors.

Several experiments were carried out during the evaluation by varying a number of parameters: the choice of what words the dimensions of the word vectors should represent (*local* vs. *global* feature selection), local feature selection according to frequency vs. χ^2 distribution, term cooccurrence context vectors vs. SVD-reduced context vectors and the number of clusters to divide the Word Space into (2 or 10). Local feature selection focuses on

the context of the ambiguous words and global feature selections chooses the n most frequent words in the corpus.

Twenty ambiguous words were chosen for disambiguation and each word was divided up into two senses, which were chosen by Schütze. There was no distinction between different parts of speech; for example, the two senses of *train* were the verb sense as in the sentence, "she will train to become a chef," and the noun sense as in the sentence, "the train got derailed." The sense labeling task was also carried out manually; that is, the author hand labeled the dictionary senses that he thought each cluster corresponded to. For the experiments where ten clusters were created, accuracy was only evaluated for two senses; the instances of the ambiguous words in the training set were also labeled so that they were assigned to a particular cluster, so these experiments were not unsupervised.

The best results came from using SVD, global feature selections and ten clusters at 90.6% average accuracy, although it achieved 89.7% average accuracy using the same setup using two clusters. The average baseline of assigning the most common sense was 61.2%. However, below baseline performance occurred in a number of cases.

These experiments evaluate how well the clusters correspond to the dictionary senses chosen by Schütze, even though the algorithm had no knowledge of such senses beforehand. Therefore, it has no guidance of what the sense groupings should be, other than the contexts. More specifically, these experiments test the following hypothesis:

> *Contextual Hypothesis for Senses:* Two occurrences of an ambiguous word belong to the same sense to the extent that their contextual representations are similar. (Schütze, 1998)

The *Contextual Hypothesis for Senses* depends heavily on the choice of features for the contextual representation; in this paper, local context was treated as a *bag of words*, without reference to syntactic dependencies, semantics, parts of speech or even word order. Better partitioning of the word space could probably be achieved by taking more features into account.

It is doubtful that an algorithm that does word sense discrimination without being guided by a set of senses will be able to create clusters that map correctly to the senses in any existing lexicon (i.e., solve the sense labeling problem). Therefore, this algorithm cannot be used for any problem that requires a mapping to senses in a lexicon; for example, word sense disambiguation can benefit machine translation only if the system knows which sense the word corresponds to so that it can choose the correct translation.

However, machine translation using this method might still be possible if cross-language correspondences between clusters could be created.

However, context-group discrimination can be used for problems where a mapping to external senses is not needed. One such problem they present is document-query similarity in Information Retrieval (IR): determining which documents a search engine should return based on how similar the query is to the documents. In a previous work (Schütze and Pedersen, 1995), they summarize an experiment using a similar system where documents and queries were represented as vectors and words were disambiguated using context-group discrimination. Documents that contained senses of the word different from the one specified in the query were filtered out.

3 Pedersen and Bruce (1998)

The goal of (Pedersen and Bruce, 1998) is the same as that of (Schütze, 1998): to attribute a sense group to each instance of some target polysemous word, where these sense groups are automatically inferred from unannotated corpora.

Given a set S of sentences that contain target polysemous word w, the algorithm converts each sentence $s \in S$ into a feature vector $(F_1, F_2, ..., F_n, S)$, where F_i, $1 \leq i \leq n$, represents a feature of the local context of w and S is the sense group that w belongs to. They use the Naive Bayes model, which when applied to this task defines the joint probability of observing features $F_1, F_2, ..., F_n$ with sense S as

$$p(F_1, F_2, ..., F_n, S) = P(S)P(F_1|S)P(F_2|S)...P(F_n|S) \quad (1)$$

Given the *parameters of the model* $P(S)$ and $P(F_i|S)$, it is possible to calculate the *sufficient statistics*: the frequencies with which F_i was observed with S. Conversely, given the sufficient statistics, it is possible to calculate the model parameters.

They test two different algorithms to estimate the parameters: Estimation-Maximization (EM) for Naive Bayes and Gibbs Sampling. These algorithms are used to impute a sense group for the missing data and estimate the parameters. It appears that they attempt to find a sense group such that $p(F_i|S)$ is maximized. EM has an estimation step (E-step) and a maximization step (M-step). Initially, $p(F_i|S)$ is initialized to a random value. Then, the E-step calculates the expected values of the sufficient statistics $count(F_i, S)$ based on the current model parameter estimates $p(S)$ and $p(F_i|S)$. Following this, the M-step calculates the parameter estimates based

on the current values of the sufficient statistics. The E-steps and M-steps continue to alternate until the parameter estimates converge. This algorithm can converge to a local maxima when the likelihood function is not well approximated by the normal distribution.

Gibbs sampling is the other method they used to estimate the parameters, which generates random samples from a probability distribution when it is difficult to sample directly from that distribution. The sampler generates chains of values for the missing senses S and model parameters $p(F_i|S)$ in an iterative process, and these chains of values will eventually converge.

There different feature sets A, B and C were defined for each word to be disambiguated and the two different algorithms EM and Gibbs were tested for each feature set. Parts of speech were not mixed for the different senses of each word, but nouns, adjectives and verbs were disambiguated. All words were divided up into two or three senses. The data in the training corpus was sense-tagged, but the senses were only used for automating the evaluation and not for learning. The results of the experiments, like in (Schütze, 1998), determine how well the automatically created sense groups line up with the predetermined sense distinctions for each word.

They record several different types of features of the local context. Feature M represents the morphology of the ambiguous word. For nouns M can only have two different values, which indicate whether the noun is plural or singular; for verbs, M indicates the tense and can have up to seven possible values. The features PL_i and PR_i, $i = 1$ or 2, represents the part of speech of the word i positions to the left or to the right, respectively, of the ambiguous word; there are five possible values for these features. Features C_i, $i = 1$, 2 or 3, can have two values and represent whether the i^{th} most frequent word in the corpus occurs in the sentence being processed. Features UL_i, $i = 1$ or 2, represent the i^{th} word occurring to the left or to the right, respectively, of the ambiguous word. UL_i features can only take on 21 possible values: 19 of the most frequent words that occur in that fixed position in all sentences that contain the ambiguous word, value (*none*) indicates that the word in position i is not among the 19 words, or value (*null*) that indicates position i is out of sentence bounds. Features CL_i and CL_i, $i = 1$, can take on the same 21 different values as features UL_i and indicate the content word occurring i positions to the left or to the right of the ambiguous word, respectively.

Feature set A combines features M, PL_1, PL_2, PR_1, PR_2, C_1, C_2 and C_3. Feature set B combines features M, UL_1, UL_2, UR_1 and UR_2. Feature set C combines features M, PL_1, PL_2, PR_1, PR_2, CL_1 and CR_2. For adjectives and verbs, both algorithms and all feature sets achieved below baseline

accuracy, and for nouns both algorithms and all feature sets achieved above baseline accuracy, but results were still in the low 60% range.

Compared to (Schütze, 1998), these algorithm seem to have faired worse, although the experiments were different and so a fair comparison cannot be made. The collocation features in this algorithm were first-order; that is, it simply recorded whether or not certain words coccourred.

4 Yarowsky (1995)

There is an unsupervised noun sense disambiguation algorithm by (Yarowsky, 1995) that trains on unannotated corpora and is able to map automatically to course-grained dictionary senses. The fact that it is able to solve the sense-labeling problem is what makes this algorithm disambiguation rather than discrimination.

A small amount of knowledge is needed to bootstrap the algorithm. For each sense of each noun to be disambiguated, it requires some *seed collocate terms*. Seed terms are words that tend to collocate with a particular sense of the noun; for example, we would expect that the seed collocate term *manufacturing* cooccurs with the industrial sense of *plant* more often than it does with the living-being sense of *plant*. The algorithm finds all sentences that contain both the ambiguous word and one of the seed collocate terms, and classifies the sense of the ambiguous word according to the sense that the seed term is associated with. Then, it attempts to find new collocate terms based on the previous sentences to repeat the process and classify new senses. These seed terms may be manually entered or automatically discovered using dictionary definitions by choosing words that appear with greater frequency relative to the rest of the dictionary.

There is an underlying assumption stating that given any particular collocation, such as *harvest plants* or *manufacturing* within a few words from *plant*, the sense of the noun will almost always be the same. This mostly-reliable heuristic is called *One Sense per Collocation*. Another heuristic that the paper uses is *One Sense per Discourse*; it states that if we have a discourse that has many uses of a particular sense of a noun that we have a high certainty about, then the other uses of that noun in that discourse probably have that same sense. Note that one example of One Sense per Discourse not helping is the collocation *green plants*, where *green* is used to mean environmentally friendly.

The paper demonstrates the algorithm on the two primary senses of the noun *plant*: the manufacturing sense and the flora sense. First, using an

untagged corpus, it identifies all local contexts containing *plant* and stores them in an initial training set. The ambiguous terms in the portion of the sentences that contain one of the seed terms are tagged with the sense associated with that seed term.

A decision-list algorithm uses the contexts of the initial tagged homographs to discover new collocate terms that reliably partition the tagged data into the two senses. Syntactic information is also recorded about the new collocate terms to differentiate between cases like *pesticide plant* and *plant pesticide*. The collocations are also ranked by how characteristic they are of a given sense.

In a similar fashion to the above, using the newly learned collocate terms, the decision-list looks for more sentences in the untagged portion of the training set and tags them with the correct sense. These sentences contain more collocations, which it learns to associate with a particular sense in the same way as previously described.

Optionally, it exploits the tendency of human language to feature only one noun homograph sense per discourse. For example, if *plant* has been tagged four times as the manufacturing sense in a discourse, and if a fifth usage of *plant* is untagged or has been tagged as the flora sense with lower confidence, then the former label will be set to the manufacturing sense. However, given a slightly different scenario where it is too close to tell which is the dominant sense, all five *plant* instances would have their labels removed. This helps prevent incorrect evidence being attributed to a particular sense and skewing the algorithm.

The decision-list algorithm and the optional one-sense-per-discourse constraint are repeated iteratively until the algorithm converges on a stable residual (unlabeled) set. Note that the decision-list algorithm uses only the single most reliable piece of evidence (collocated term plus syntactic features) which it associates with a given sense, and not a combination of of different pieces of evidence. Therefore, the correctness of a given sense label hinges on the correctness of a particular collocate term. If newly acquired evidence discredits the correctness of said term, that term will have its label removed; it will usually be relabeled in future iterations and associated with more distinguishing evidence. At this point, the algorithm may be used to automatically tag new sentences with one of the two senses of *plant*.

The algorithm was trained on a 460 million word corpus. It achieved 98.6% accuracy for *plant* and slightly lower for less distinguished terms, such as 93.6% for *space* in the volume and outer-space senses and 93.9% for *drug* in the medicine and recreational senses. The results were also compared to a decision-list supervised training algorithm, which were slightly lower.

This paper shows that unsupervised nouns sense disambiguation can be done with very high accuracy. However, it would be interesting to see how the algorithm would perform when trying to distinguishing between more than two senses or senses that are not as well distinguished. For example, *bank* can mean a physical building, the funds held by a gambling house or dealer, a money box or an institution. All of these are closely associated with the concept of money. It might have some trouble with this, as it did have lower accuracy when trying to disambiguate between the two senses of *drug*; these senses of *drug* are problematic even for humans.

5 Bordag (2006)

(Bordag, 2006) introduces a triplet-based clustering WSI algorithm that is based on the one sense per collocation heuristic of (Yarowsky, 1995) and uses an automatic pseudoword evaluation method, similar to (Schütze, 1998). The idea behind the algorithm is that given a word w and its local context, w and two of its content collocate terms, w_i and w_j, together tend to uniquely identify some sense or topic. Let v_w denote the set of the 200 most significant cooccurrences of w in the corpus. Then for all possible triplets (w, w_i, w_j), $K = v_w \cap v_{w_i} \cap v_{w_j}$ is a feature of the local context of w. All such K's for all instances of w are clustered and the clusters represent the different senses of w.

As an example, consider the sentence "NASA wants a space mission to Mars," where *space* is the target word, and consider the particular triplet *(space, MARS, NASA)*. One of the features of this local context would be $K = v_{space} \cap v_{MARS} \cap v_{NASA}$, which might look something like *(launch, probe, cosmonaut, ...)*. Similar K's would be created for the other combinations of triplets in this and all local contexts where *space* appears. Ideally, the K's for the instances where *space* is being used in the *outer space* term will be clustered into a different set from the K's for the instances where *space* is being used in the *geometrical* sense.

This paper pointed out that there are two types of ambiguity: syntactic and semantic. The main syntactic ambiguity is the ambiguity between the different parts of speech: nouns, verbs, adjectives, adverbs, etc. Semantic ambiguity is the ambiguity between different word meanings. Therefore, it is possible that clusters which make the distinctions between syntactic or semantic ambiguity will get created. They hypothesized that using shorter windows would lead to words being clustered more by syntactic ambiguity and larger window sizes would lead more to semantic ambiguity. Their

results were inconclusive because they got very bad results that they did not go into detail about for using short window sizes; they suggested this could be due to data sparseness.

The evaluation technique used by this paper as well partially by (Schütze, 1998) is pseudoword evaluation. This is a method of choosing two words from the corpus, say *banana* and *door*, and creating a new pseudoword *bananadoor*. This pseudoword can be interpreted as having two senses: *banana* and *door*. All instances of both *banana* and *door* are then replaced by *bananadoor* in the corpus. The WSI algorithm is run and a number of sense clusters are created for *banandoor*; note that in general, it is possible that more than two sense clusters are created if one or both of the words making up the pseudoword is polysemous but in this case *banana* and *door* are assumed to be monosemous. For example, we would expect that a cluster containing mostly the contexts of *door* is created (C_{door1}) and a cluster containing mostly the contexts of *banana* is created ($C_{banana1}$). We can return to the original corpus and check the overlap between the cluster created by *door* and one of the clusters created by C_{door1}. F-measure was 78.66% using triplets and 72..61% using pairs.

This paper is similar (Yarowsky, 1995) where a word and one of its collocate terms identify a sense. But in that paper, certain collocate relationships were stronger than others (such as predicate-argument) and it was only certain words that tended to favor one sense over the other. For example, *a plant* does not suggest a particular sense of *plant*. Unlike (Yarowsky, 1995), (Bordag, 2006) does not record the specific type of collocation; that is, it treats the local context like a bag of words, not using the syntactic properties or even word positions as features.

6 Purandare and Pedersen (2004)

(Purandare and Pedersen, 2004) systematically compares the approaches by (Schütze, 1998) and (Pedersen and Bruce, 1997). The latter approach was presented above and used vector spaces using second order cooccurrence context representation. The former was not summarized here, but uses similarity spaces and first order cooccurrence context representation. The idea of (Pedersen and Bruce, 1997) is to represent first order contexts in a similarity space where each context can be thought of as a point and the weights of the edges between the points are a function of the similarity. They found that second order context vectors perform better when data is sparse.

7 Conclusion

This paper has summarized and discussed several papers on WSI as well as (Yarowsky, 1995), which is a knowledge-lean approach to WSD. Typically these methods attempt to be knowledge independent and also tend to use the bag of words model when characterizing the contexts. They rely on the assumption that the context is sufficient to induce a set of senses that corresponds to the sense distinctions in a dictionary. Evaluation of these methods test this hypothesis. Typically these methods do not perform well above baseline and can only handle course-grained senses, which may suggest that the bag of words model that is usually employed may not be enough. (Schütze, 1998) suggested one possible application in IR that did not have to work with an external set of senses.

References

Stefan Bordag. Word sense induction: Triplet-based clustering and automatic evaluation, 2006. URL `citeseer.ist.psu.edu/bordag06word.html`.

Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, Massachusetts, 1998.

T. Pedersen and R. Bruce. Knowledge lean word sense disambiguation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 800–805, Madison, WI, July 1998. URL `citeseer.ist.psu.edu/article/pedersen98knowledge.html`.

Ted Pedersen and Rebecca Bruce. Distinguishing word senses in untagged text. In Claire Cardie and Ralph Weischedel, editors, *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 197–207. Association for Computational Linguistics, Somerset, New Jersey, 1997. URL `citeseer.ist.psu.edu/552289.html`.

Amruta Purandare and Ted Pedersen. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of CoNLL-2004*, pages 41–48, 2004.

Hinrich Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124, 1998. URL `citeseer.ist.psu.edu/article/schutze98automatic.html`.

Hinrich Schütze and Jan O. Pedersen. Information retrieval based on word senses, 1995. URL citeseer.ist.psu.edu/sch95information.html.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Meeting of the Association for Computational Linguistics*, pages 189–196, 1995. URL citeseer.ist.psu.edu/yarowsky95unsupervised.html.