An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet

Satanjeev Banerjee and Ted Pedersen

University of Minnesota, Duluth, MN 55812 USA
{bane0025, tpederse}@d.umn.edu
http://www.d.umn.edu/{~bane0025,~tpederse}

Abstract. This paper presents an adaptation of Lesk's dictionary–based word sense disambiguation algorithm. Rather than using a standard dictionary as the source of glosses for our approach, the lexical database WordNet is employed. This provides a rich hierarchy of semantic relations that our algorithm can exploit. This method is evaluated using the English lexical sample data from the SENSEVAL-2 word sense disambiguation exercise, and attains an overall accuracy of 32%. This represents a significant improvement over the 16% and 23% accuracy attained by variations of the Lesk algorithm used as benchmarks during the SENSEVAL-2 comparative exercise among word sense disambiguation systems.

1 Introduction

Most words in natural languages are *polysemous*, that is they have multiple possible meanings or *senses*. For example, *interest* can mean a charge for borrowing money, or a sense of concern and curiosity. When using language humans rarely stop and consider which sense of a word is intended. For example, in *I have an interest in the arts*, a human reader immediately knows from the surrounding context that *interest* refers to an appreciation, and not a charge for borrowing money.

However, computer programs do not have the benefit of a human's vast experience of the world and language, so automatically determining the correct sense of a polysemous word is a difficult problem. This process is called *word sense disambiguation*, and has long been recognized as a significant component in language processing applications such as information retrieval, machine translation, speech recognition, etc.

In recent years corpus–based approaches to word sense disambiguation have become quite popular. In general these rely on the availability of manually created *sense–tagged* text, where a human has gone through a corpus of text, and labeled each occurrence of a word with a tag that refers to the definition of the word that the human considers most appropriate for that context. This sense–tagged text serves as training examples for a supervised learning algorithm that can induce a classifier that can then be used to assign a sense–tag to previously unseen occurrences of a word. The main difficulty of this approach is that

sense–tagged text is expensive to create, and even once it exists the classifiers learned from it are only applicable to text written about similar subjects and for comparable audiences.

Approaches that do not depend on the existence of manually created training data are an appealing alternative. An idea that actually pre–dates most work in corpus–based approaches is to take advantage of the information available in machine readable dictionaries. The Lesk algorithm [3] is the prototypical approach, and is based on detecting shared vocabulary between the definitions of words. We adapt this algorithm to WordNet [2], which is a lexical database structured as a semantic network.

This paper continues with a description of the original Lesk algorithm and an overview of WordNet. This is followed by a detailed presentation of our algorithm, and a discussion of our experimental results.

2 The Lesk Algorithm

The original Lesk algorithm [3] disambiguates words in short phrases. The definition, or *gloss*, of each sense of a word in a phrase is compared to the glosses of every other word in the phrase. A word is assigned the sense whose gloss shares the largest number of words in common with the glosses of the other words. For example, in *time flies like an arrow*, the algorithm compares the glosses of *time* to all the glosses of *fly* and *arrow*. Next it compares the glosses of *fly* with those of *time* and *arrow*, and so on. The algorithm begins anew for each word and does not utilize the senses it previously assigned.

The original Lesk algorithm relies on glosses found in traditional dictionaries such as Oxford Advanced Learner's. We modify Lesk's basic approach to take advantage of the highly inter–connected set of relations among synonyms that WordNet offers. While Lesk's algorithm restricts its comparisons to the glosses of the words being disambiguated, our approach is able to compare the glosses of words that are related to the words to be disambiguated. This provides a richer source of information and improves overall disambiguation accuracy. We also introduce a novel scoring mechanism that weighs longer sequences of matches more heavily than single words.

3 About WordNet

While traditional dictionaries are arranged alphabetically, WordNet is arranged semantically, creating an electronic lexical database of nouns, verbs, adjectives, and adverbs. Synonymous words are grouped together to form synonym sets, or *synsets*. A word is polysemous if it occurs in several synsets, where each synset represents a possible sense of the word. For example *base* occurs in two noun synsets, {*base, alkali*} and {*basis, base, foundation, fundament, groundwork, cornerstone*}, and the verb synset {*establish, base, ground, found*}.

In WordNet version 1.7 there are 107,930 nouns arranged in 74,448 synsets, 10,860 verbs in 12,754 synsets, 21,365 adjectives in 18,523 synsets, and 4,583

adverbs in 3,612 synsets. Function words such as *for*, *the*, *and*, etc. are not defined in WordNet. Our algorithm only disambiguates words that belong to at least one synset, which we call *WordNet words*.

Each synset has an associated definition or gloss. This consists of a short entry explaining the meaning of the concept represented by the synset. The gloss of the synset {*base, alkali*} is "any of various water-soluble compounds capable of turning litmus blue and reacting with an acid to form a salt and water", while that associated with {*basis, base, foundation, fundament, groundwork, cornerstone*} is "lowest support of a structure". Each synset can also be referred to by a unique identifier, commonly known as a *sense–tag*.

Synsets are connected to each other through a variety of semantic relations. With few exceptions these relations do not cross part of speech boundaries, so synsets are only related to other synsets that belong to the same part of speech. Here we review only those relations that have entered into the experiments presented in this paper. A complete description of all the relations can be found in [2].

For nouns, two of the most important relations are *hyponymy* and *hypernymy*. If synset A is *a kind of* synset B, then A is the hyponym of B, and B is the hypernym of A. For example, {*bed*} is a hyponym of {*basis, base, foundation, fundament, groundwork, cornerstone*}, and conversely, {*basis, base, foundation, fundament, groundwork, cornerstone*} is the hypernym of {*bed*}. Another pair of related relations for nouns is that of *holonymy* and *meronymy*. Synset A is a meronym of synset B if A *is a part of* B. Conversely, B is a holonym of A if B *has* A *as a part*. Thus {*structure, construction*} is a meronym of {*basis, base, foundation, fundament, groundwork, cornerstone*}, and {*basis, base, foundation, fundament, groundwork, cornerstone*} is a holonym of {*structure, construction*}.

Verbs are related through the relations *hypernymy* and *troponymy*. Synset A is the hypernym of B, if B *is one way to* A; B is then the troponym of A. Thus, the verb synset {*station, post, base, send, place*} is the troponym of {*move, displace*} since to {*station, post, base, send, place*} is one way to {*move, displace*}.

One of the few relations available for adjectives is *attribute* that relates an adjective to a noun. For example, the attribute of {*beautiful*} is the noun {*beauty*}. This is an unusual relation, in that it crosses part of speech boundaries to connect an adjective synset with a noun synset.

4 The Adapted Lesk Algorithm

This algorithm takes as input an example or *instance* in which a single *target word* occurs, and it will output a WordNet sense for that target word based on information about the target word and a few immediately surrounding words that can be derived from WordNet.

Our experimental data is the English lexical sample from SENSEVAL-2, where each instance of a target word consists of the sentence in which it occurs, along

with two or three surrounding sentences. However, our algorithm utilizes a much smaller window of context that surrounds the target word.

We define the *context* of the target word to be a window of n WordNet word tokens to the left and another n tokens to the right, for a total of $2n$ surrounding words. We include the target word in the context as well, giving a total context size of $2n + 1$ word tokens. Repeated occurrences of a WordNet word in the window are treated separately.

If the target word is near the beginning or end of the instance, we add additional WordNet words from the other direction. This is based on the suggestion of Lesk [3] that the quantity of data available to the algorithm is one of the biggest factors to influence the quality of disambiguation. We therefore attempt to provide roughly the same amount of data for every instance of every target word.

4.1 Definitions

Let the size of window of context, $2n + 1$ be designated by N. Let the WordNet words in the window of context be designated as W_i, $1 \leq i \leq N$. If the number of WordNet words in the instance is less than $2n + 1$, all of the WordNet words in the instance serve as the context.

Each word W_i has one or more possible senses, each of which is represented by a unique synset having a unique sense–tag. Let the number of sense–tags of the word W_i be represented by $|W_i|$. Hereafter we use sense–tag to refer to a sense of a word.

We evaluate each possible combination of sense–tag assignments for the words in the context window. There are $\prod_{i=1}^{N} |W_i|$ such combinations, each of which we refer to as a *candidate combination.*

A *combination score* is computed for each candidate combination. The target word is assigned the sense–tag of the candidate combination that attains the maximum score. While this combination also provides sense tags for the other words in the window of context, we view these as a side effect of the algorithm and do not attempt to evaluate how accurately they are disambiguated.

4.2 Processing

This algorithm compares glosses between each pair of words in the window of context. If there are N words in the window of context then there are $N(N-1)/2$ pairs of words to be compared. There are a series of *relation pairs* that identify which synset is to provide the gloss for each word in a pair during a comparison. For example, a relation pair might specify that the gloss of a synset of one word is to be compared with the gloss of a hypernym of the other word. The glosses to be compared are those associated with the senses given in the candidate combination that is currently being scored.

In our experiments, we compare glosses associated with the synset, hypernym, hyponym, holonym, meronym, troponym, and attribute of each word in the pair. If the part of speech of a word is known, as is the case for target words, then we restrict the relations and synsets to those associated with that part of speech. If the part of speech is not known, as is the case for the other words in the context, then we use relations and synsets associated with all the possible parts of speech. Since there are 7 possible relations, there are at most 49 possible relation pairs that must be considered for a particular pair of words. However, if we know the part of speech of the word, or if the word is only used in a subset of the possible parts of speech, then the number of relation pairs considered is less. The algorithm is not dependent on any particular relation pairs and can be run with as many or as few as seems appropriate.

When comparing two glosses, we define an *overlap* between them to be the longest sequence of one or more consecutive words that occurs in both glosses. Each overlap found between two glosses contributes a score equal to the square of the number of words in the overlap.

Two glosses can have more than one overlap where each overlap covers as many words as possible. For example, the sentences *he called for an end to the atrocities* and *after bringing an end to the atrocities, he called it a day* have the following overlaps: *an end to the atrocities* and *he called*. We stipulate that an overlap not be made up entirely of *non–content* words, that is pronouns, prepositions, articles and conjunctions. Thus if we have *of the* as an overlap, we would ignore it.

Once all the gloss comparisons have been made for every pair of words in the window based on every given relation pair, we add all the individual scores of the comparisons to arrive at the combination score for this particular candidate combination of sense–tags. This process repeats until all candidate combinations have been scored.

The candidate combination with the highest score is the winner, and the target word is assigned the sense given in that combination. In the event of a tie between two candidate combinations we choose the one that has the most familiar sense for the target word, as specified by WordNet.

5 Empirical Evaluation

We have evaluated this algorithm using the test data from the English lexical sample task used in the SENSEVAL-2 comparative evaluation of word sense disambiguation systems. The 73 target words in this data are listed below. There are a total of 4,328 test instances, divided among 29 nouns, 29 verbs, and 15 adjectives. Each word is followed by the accuracy attained by our algorithm, the number of possible WordNet senses, and the number of test instances. Note that accuracy is defined to be the number of correctly disambiguated instances divided by the number of total test instances for a word.

Nouns: art (0.500, 4, 98), authority (0.337, 7, 92), bar (0.113, 17, 151), bum (0.178, 6, 45), chair (0.522, 6, 69), channel (0.096, 10, 73), child (0.500, 4, 64), church (0.453, 4, 64), circuit (0.247, 7, 85), day (0.172, 10, 145), detention (0.625, 2, 32), dyke (0.286, 3, 28), facility (0.293, 5, 58), fatigue (0.279, 6, 43), feeling (0.275, 6, 51), grip (0.078, 11, 51), hearth (0.562, 3, 32), holiday (0.710, 3, 31), lady (0.566, 3, 53), material (0.217, 11, 69), mouth (0.400, 11, 60), nation (0.730, 4, 37), nature (0.370, 5, 46), post (0.203, 20, 79), restraint (0.200, 6, 45), sense (0.377, 6, 53), spade (0.273, 4, 33), stress (0.256, 8, 39), yew (0.607, 2, 28)

Accuracy for nouns = 0.322, 564 of 1754 correct

Verbs: begin (0.475, 11, 280), call (0.091, 41, 66), carry (0.091, 40, 66), collaborate (0.900, 2, 30), develop (0.261, 21, 69), draw (0.049, 44, 41), dress (0.220, 19, 59), drift (0.062, 17, 32), drive (0.167, 33, 42), face (0.237, 23, 93), ferret (1.000, 5, 1), find (0.029, 18, 68), keep (0.164, 25, 67), leave (0.288, 17, 66), live (0.313, 19, 67), match (0.238, 18, 42), play (0.197, 53, 66), pull (0.033, 25, 60), replace (0.289, 4, 45), see (0.159, 26, 69), serve (0.118, 16, 51), strike (0.056, 26, 54), train (0.286, 17, 63), treat (0.409, 9, 44), turn (0.060, 38, 67), use (0.658, 13, 76), wander (0.100, 5, 50), wash (0.167, 19, 12), work (0.083, 34, 60)

Accuracy for verbs = 0.249, 450 of 1806 correct

Adjectives: blind (0.782, 10, 55), colourless (0.400, 2, 35), cool (0.403, 11, 52), faithful (0.783, 5, 23), fine (0.443, 15, 70), fit (0.448, 16, 29), free (0.378, 20, 82), graceful (0.793, 2, 29), green (0.404, 15, 94), local (0.289, 5, 38), natural (0.262, 13, 103), oblique (0.345, 3, 29), simple (0.500, 9, 66), solemn (0.920, 2, 25), vital (0.632, 4, 38)

Accuracy for adjectives = 0.469, 360 of 768 correct

Thus, overall accuracy is 31.7%, where 1374 of 4328 test instances are disambiguated correctly. In SENSEVAL-2 two variations of the Lesk algorithm were provided as benchmarks. The first counts the number of words in common between the instance in which the target word occurs and its gloss, where each word count is weighted by its inverse document frequency. Each gloss is considered a separate document in this approach. The gloss with the highest number of words in common with the instance in which the target word occurs represents the sense assigned to the target word. This approach achieved 16% overall accuracy. A second approach proceeded identically, except that it added example texts that WordNet provides to the glosses. This achieved accuracy of 23%. Since our approach does not use example texts, the most indicative comparison is with the first approach. Thus, by including an extended notion of which glosses to compare a target word's gloss with, we have doubled the accuracy from 16% to 32%. The fact that the example texts provided by WordNet improved the accuracy of these benchmark approaches suggests that we should consider using this information as well.

In addition, our approach compares favorably with other systems entered in SENSEVAL-2. Of the seven unsupervised systems that did not use any of the available training examples and only processed test data, the highest ranked

achieved accuracy of 40%. There were four approaches that achieved accuracy of less than 30%.

6 Analysis of Results

In preliminary experiments we ignored the part of speech of the target word, and we included overlaps that consist entirely of non–content words. While the overall accuracy of this approach was only 12%, the accuracy for the nouns was 29%, which is only slightly less than that obtained when using the part of speech information for target words. However, significant reductions in accuracy were observed for adjectives (11%) and verbs (7%).

These results confirm the notion that WordNet is a particularly rich source of information about nouns, especially when considering the hypernym and hyponym relations. When compared to verbs and adjectives, there is simply more information available. When we ignored part of speech distinctions in the target word, those that can be used in multiple parts of speech such as *dress, blind,* etc., made gloss comparisons involving all their possible parts of speech. In doing so, they were likely overwhelmed by the sheer volume of noun information, and this resulted in poor accuracy when the target word was in fact an adjective or verb.

This algorithm very rarely encounters situations where it can not make a determination as to sense–tags. A candidate combination with no overlaps receives a score of zero. If every candidate combination associated with a particular target word gets a score of zero, then the algorithm assigns every word in the window with its most familiar sense, according to WordNet. However, there were only ten test instances for which our algorithm had to resort to this default. If there are two or more candidate combinations tied at the highest score, then we report the most familiar of these senses. Such ties are also rare, occurring only 57 times out of 4,328 test instances.

7 Discussion

There are numerous issues that arise in formulating and refining this algorithm. We discuss a few of those issues here, among them how to represent context, which relations should be the basis for comparisons, how to score matches, and how to deal with possible performance issues. The current approach is a first approximation of how to use a Lesk algorithm with WordNet, so certainly there is room for considerable variation and experimentation.

7.1 Context

Our choice of small context windows is motivated by Choueka and Lusignan [1], who found that human beings make disambiguation decisions based on very short windows of context that surround a target word, usually no more than two

words to the left and two words to the right. While WordNet does not provide anywhere near the same level of knowledge about words that a human being has, it encodes at least a portion of such information through its definitional glosses and semantic relations between synsets. We believe this provides sufficient information to perform word sense disambiguation across a range of words and parts of speech.

For example, consider the sentence *I put money in the bank*. A human being asked to disambiguate *bank* knows that it is much more common to put money into a financial institution rather than a slope. WordNet supports the same inference if one observes that a *bank* "channels... money into lending activities" when it is a *financial institution* and not when it is a *slope*. The fact that *money* occurs in the definition of one sense and not in the other implies a strong connection between money and that sense of bank. Given that the words in a sentence usually have a flow of related meanings, it is very likely that successive words in a sentence will be related.

By identifying overlaps between the senses of one word and those of the next in a context window, we are trying to identify any such connection between a particular sense of one word and that of the next. Such connections are not accidental but are indicative of the senses in which these words are used.

7.2 Relations

This algorithm depends very much on the relation pairs that are employed. We have only experimented with synsets that are directly related to the words being compared, however, other more indirect relations may provide useful information. One example is the coordinate or sister relation, which consists of the hyponyms of the hypernym of a synset.

As was mentioned earlier, we have not used every relation provided in WordNet. Among those left out are *cause* and *entailment* for verbs and *similar to*, *participle of* and *pertainym of* for adjectives. There is also an *antonymy* relation that applies to all parts of speech, that relates a synset to another that represents its opposite in meaning. This is particularly intriguing in that it provides a source of negative information that will allow our algorithm to identify the sense of a word based on the absence of its antonymous sense in the window of context.

A single synset may be related to multiple synsets through a single relation. For example, when a relation pair includes hyponomy, we concatenate the glosses of all the hyponym synsets and treat them as one during matching. We do not distinguish between separate synsets, as long as they are all related to the synset through the same relation.

Our scoring mechanism also does not distinguish between matches amongst different relations; all relation–pairs are treated equally. Thus an n word match between two hypernyms gets precisely the same score as an n word match between a hyponym and a hypernym. As yet we have no reason to prefer matches between certain pairs of relations over others, and so the scoring is uniform.

As we begin to better understand which relation pairs lead to more meaningful matches, we will likely adjust the scoring mechanism to reward more useful matches.

7.3 Scoring

One of the novel aspects of this approach is the fact that scores are based on the length of the match. By using the square of the number of words in the match as the score, we appeal to Zipf's Law which states that the frequency of an event is inversely proportional to the rank of that event, where the ranking is based on the frequencies of all events. This implies that most events occur only once, and only a few occur with greater frequency. The occurrence of individual words in a corpus of text holds to this distribution, and it also applies to the occurrence of multi-word sequences. As word sequences grow longer, it is increasingly rare to observe them multiple times in a corpus. Thus, if we find a multi-word match between two glosses, this is a remarkable event, and merits more than a linear increase in scoring. By squaring the length of the match, we give a higher score to a single n word sequence than to the combined score of those n words, if they were to occur in shorter sequences.

Partial word matches are discarded so as to rule out spurious matches. For example between *Every dog has its day* and *Don't make hasty decisions*, there exists an overlap *has*, which is not particularly useful. We also discard overlapping sequences that consist entirely of function words since these are also of questionable value and may skew results towards longer glosses that happen to contain more function words. However, sequences of content words that also contain function words are retained. This is to preserve longer sequences, as opposed to breaking them down into smaller sequences due to the presence of function words. In future we will consider *fuzzier matching* schemes, where stemming or measures of edit distance are employed to account for near matches. Sidorov and Gelbukh [4] present such an approach in a variant of the Lesk algorithm applied to a Spanish explanatory dictionary.

In scoring a combination of candidate senses, we compare all pairs of words in the context window. Thus, if we have a five word context window, a strong relationship between the words on the extreme ends can force a certain sense for the words in the middle of the window. A possible variation suggested by Lesk [3] is to weigh the score of a pair of words by the distance between them. Thus, one might give higher scores to words that appear more closely together in the window of context.

7.4 Performance

Each WordNet word usually has multiple sense–tags. As the window of context becomes larger, the number of possible combinations of candidate sense–tags grows rapidly. There are three immediate courses of action that we can take to alleviate this problem. The first is to part of speech tag all of the words in the window of context, and thereby restrict the range of their possible sense

tags to those associated with the given part of speech. The second is to focus the algorithm strictly on the target word and eliminate all comparisons between pairs of glosses that do not involve the target word. The third is to restrict the consideration of possible senses to among the most familiar in WordNet.

8 Conclusions

This paper presents an adaptation of the Lesk algorithm for word sense disambiguation. While the original algorithm relies upon finding overlaps in the glosses of neighboring words, this extends these comparisons to include the glosses of words that are related to the words in the text being disambiguated. These relationships are defined by the lexical database WordNet. We have evaluated this approach on the English SENSEVAL-2 lexical sample data and find that it attains overall accuracy of 32%, which doubles the accuracy of a more traditional Lesk approach. The authors have made their Perl implementation of this algorithm freely available on their web sites.

9 Acknowledgments

This work has been partially supported by a National Science Foundation Faculty Early CAREER Development award (#0092784). We would like to thank the SENSEVAL-2 coordinators for putting their data in the public domain. We are grateful to Jason Rennie for making his Perl WordNet interface, QueryData, available, and to the WordNet team for their invaluable contributions to this research. And of course, we are indebted to Michael Lesk whose original work is the inspiration of this particular approach.

References

1. Y. Choueka and S. Lusignan. Disambiguation by short contexts. *Computers and the Humanities*, 19:147–157, 1985.
2. C. Fellbaum, editor. *WordNet: An electronic lexical database*. MIT Press, 1998.
3. M. Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In *Proceedings of SIGDOC '86*, 1986.
4. G. Sidorov and A. Gelbukh. Word sense disambiguation in a Spanish explanatory dictionary. In *Proceedings of TALN*, pages 398–402, Tours, France, 2001.