Overview and evaluation of an unsupervised approach to lexical substitution

Outline

Most words have multiple meanings – a fact that complicates a range of natural language processing (NLP) tasks. Successful approaches to machine translation, question answering, information retrieval, summarisation, text simplification, and more, require an ability to distinguish between the multiple possible senses of homonymous and polysemous words.

This task of computationally identifying the meaning of a word in a given context is known as word sense disambiguation (WSD). While a WSD algorithm can be assessed in terms of how far it improves the performance of one of the 'end-to-end' applications listed above, such as machine translation, intrinsic evaluations – where the WSD system is treated independently – are faster and more common (Jurafsky and Martin, 2017).

One framework for this sort of intrinsic evaluation is Diana McCarthy and Roberto Navigli's English Lexical Substitution Task, presented at Semeval-2007 (McCarthy and Navigli, 2007). The task requires systems to select appropriate substitutes for a sample of nouns, verbs, adjectives and adverbs, all appearing in the context of sentences from the English Internet Corpus of English (Sharoff, 2006). These substitues are compared to a "gold standard" of alternatives picked by human annotators, and the performance of the systems is measured accordingly. As such, it should be noted that systems could be seen to complete two tasks: identifying the sense of the target word in context, in the first instance, and selecting an appropriate alternative word in the second.

This report outlines one unsupervised approach, with a number of variations, to carrying out the Semeval task. After some detailed discussion of the method itself in its different guises, performance on the task will be evaluated and analysed, with potential improvements and alternative – likely more effective – approaches suggested.

Method

The most successful methods for tackling lexical substitution tasks tend to be 'supervised', extracting features from corpora of sense-labelled data and training a classifier to identify the correct sense on the basis of these features (Jurafsky and Martin, 2017). However, building large corpora in which words are sense-labelled is expensive. 'Semi-supervised' techniques, or 'bootstrapping', can also be effective, starting with a small set of data labelled by hand or according to a heuristic (see Yarowsky 1995) and training a classifier on the seed set, before expanding that set with any newly-labelled data that the classifier is very confident about - then training a new classifier on the expanded seed set, and so on until the data are all confidently labelled.

The method adopted here, though, is unsupervised. It is a knowledge-based approach, relying on the WordNet lexical database. Although alternatives, such as graph-based methods (see Navigli and Lapata, 2010) or – not relying on a knowledge-base at all – word sense induction (WSI) are also possible, our focus is on variations of the Lesk (1986) algorithm. Since dictionary-based methods are perhaps the most intuitive of the unsupervised approaches to WSD, and Lesk variants are the most well-studied of these (Jurafsky and Martin, 2017), this seemed a sensible place to start when trying to understand the problems facing a system attempting to disambiguate word senses. While outstanding results were not expected, it was hoped that attempting to implement Lesk variants would at least confirm their limitations, and help point the way towards likely more successful approaches.

The implementation detailed below takes inspiration more from what is known as the 'Simplified Lesk' algorithm (Vasilescu et al., 2004) than it does from the original. Where the original compared the dictionary glosses and examples of the target word with the corresponding signatures of each of the context words, this approach looks only to find crossover between the target's signature and the context words themselves. Using WordNet, the theory is that the sense of a word which shares the most non-stopwords in its description and examples with the sentence the target word appears in ought to be the appropriate sense. If no WordNet synset is identified as corresponding to the most appropriate sense – if, in other words, no crossover is found between any of the synset signatures and the other words in the sentence – the algorithm follows Vasilescu et al. (2004) in selecting the first and most frequent synset.

With a most likely sense identified, a substitute word is then picked from the alternative lemmas associated with that sense. Note that only one substitute is returned: although the human annotators on the task were allowed to select up to three words, and the original evaluations outlined by McCarthy and Navigli (2007) for the task involved systems picking multiple possible alternatives for a word, this approach does not seem hugely helpful for applications in the real world. It seemed more sensible to return one alternative, and see whether that appeared on the human-annotated list. If there are no alternative lemmas associated with the selected synset, the algorithm picks the most appropriate hypernym of the synset and selects a lemma from there ('container' might be selected to replace 'tin', for instance).

Parameter Settings

A number of optional extensions and tweakable parameters are incorporated into the algorithm. These parameters are intended to test the impact of different gloss sizes and types on the performance of dictionary-based Lesk, and the effect of different approaches to identifying an alternative lemma once the sense of the word has been disambiguated.

The first parameter choice is between a most-frequent-sense approach, which naïvely takes the first WordNet synset for the target word as its sense, and making use of the Lesk implementation. The former is included primarily as a baseline to compare the performance of the more complex approaches to.

The second choice is between a number of possible extensions of the Lesk algorithm, intended to assess the impact that feeding different levels of information into sense glosses has on WSD accuracy. Expanding sense glosses ought to improve performance by making overlap between glosses and the target sentence more likely, enabling an informed decision to be made as to the best synset – rather than simply selecting the most common one. The first layer of these extensions was intuitive, but follows (in part) Banerjee and Pedersen (2003). It finds the definitions and examples for any hyper- and/or hyponyms of each synset, and incorporates these into that synset's gloss.

The second extension is a part-implementation of the 'Corpus Lesk' algorithm, found by Vasilescu et al. (2004) to be the most successful of the Lesk family. In a separate piece of code, prior to running the lexical substitution algorithm itself, every sentence in the SemCor sense-labelled corpus is iterated over, in a search for words tagged with any of the synsets associated with our target words. When such a word is found, a dictionary entry is created mapping the name of the synset in question to the words in the sentence (see the get_sense_dict function in code appendix). This dictionary entry is expanded with more and more words over time, as other words labelled with that synset are

found and the context they appear in recorded. These bag-of-words dictionary entries can then be included in the gloss for any of the synsets when the lexical substitution algorithm is looking for crossover between target sentence and gloss.

Where a chosen synset offers multiple alternative lemmas, different approaches to selecting which to use are also included in the algorithm. The default is to select the first listed lemma in WordNet for a chosen synset. The alternatives are to manually check the count of each lemma in WordNet and pick the most prevalent, or to measure the similarity of each lemma against the target word and select the alternative that appears to be most similar. This similarity assessment is carried out using GoogleNews vectors with the gensim implementation of Word2Vec.

Other parameters that could be explored, but which are kept stable in this implementation, include the way that crossover between glosses and target context is measured, how that context is defined, and the lexical resources that are used. With regard to the measurement of crossover, the algorithm simply counts the number of non-stopword words that are shared between the target sentence and glosses. The Corpus Lesk algorithm as defined by Vasilescu et al. (2004), however, does not remove stopwords and instead uses a measure of inverse document frequency (IDF) to assign weights to any words that do crossover. The difference such an approach might make is not assessed here.

For this implementation, target context is defined simply as the bag of words occurring in the sentence with the target word – with stopwords and numbers removed. The impact of stemming or lemmatising these words might also have been considered, but is not. Other definitions of context could have made use of the parsed lexsub_trial file, and taken into account the dependency relationships the target word appears in, but this would have been better suited to supervised algorithms or clustering methods. The lexical resource used is also limited to WordNet (and SemCor), although alternative machine-readable thesauri or dictionaries, such as Roget's Thesaurus or Byblo, could have been exploited.

Finally, it should be noted that the algorithm only works for target words that are nouns. It could have been expanded to verbs, adjectives or adverbs, all of which are included in WordNet, with just a few tweaks – as long as the algorithm knew the part of speech (PoS) of the word in question, it could simply pass this PoS to the WordNet lemmatize() and synsets() functions and the process would work as before. There were enough difficulties with getting the algorithm to work with nouns alone, though, that such an extension did not seem necessary at this point.

Method of Evaluation

The proposed method of evaluation is very simple. Since all of the substitutes selected by human annotators should be acceptable as replacements in a real-world setting, the substitute offered by the system will be deemed correct if it is in the list of gold standard alternatives for the target word, regardless of its position in that list. The 'substitute accuracy' score presented below is therefore the number of target words out of the 73 nouns for which a substitute was returned that was also one of the alternatives selected by the human annotators.

To distinguish between failures to identify the correct sense of the target word, and failures to return a word that is on the gold standard list despite the correct sense being identified, a 'sense accuracy' score is also presented below. This is a hazier measure, insofar as the exact sense of a target word is often ambiguous (as is borne out by the

variation in some of the human annotations). For the purposes of the below, a sense will be deemed accurately identified if one of the lemmas for the selected synset appears in the gold list, even if that lemma was not the one returned. Similarly, if the lemmas attached to the selected synset carry much the same meaning as one of the words returned by the human annotators, but WordNet happens not to include said word in the list of synset lemmas, the sense will be considered accurately identified.

Splitting the evaluation into these two separate measures enables a more appropriate comparison of different gloss types and sizes for synset selection – which should primarily affect sense accuracy, and substitute accuracy only indirectly as a result – and different approaches to lemma selection – which should affect the accuracy of the substitutes returned, but not that of the senses identified. This, in turn, should allow for a better sense of what underlies the failings of the system.

Results

ID	Synset selection	Gloss type	Lemma selection	Sense accuracy (/73)	Substitute accuracy (/73)
1-1-1	First synset	N/A	First lemma	32	28
1-1-2	First synset	N/A	Highest count	32	**30**
1-1-3	First synset	N/A	Most similar	32	26
2-1-1	'Best' synset	Synset gloss	First lemma	29	25
2-1-2	'Best' synset	Synset gloss	Highest count	29	25
2-1-3	'Best' synset	Synset gloss	Most similar	29	25
2-2-1	'Best' synset	Synset + hyponyms	First lemma	26	23
2-2-2	'Best' synset	Synset + hyponyms	Highest count	26	23
2-2-3	'Best' synset	Synset + hyponyms	Most similar	26	22
2-3-1	'Best' synset	Synset + hypo/hyper	First lemma	27	24
2-3-2	'Best' synset	Synset + hypo/hyper	Highest count	27	24
2-3-3	'Best' synset	Synset + hypo/hyper	Most similar	27	23
2-4-1	'Best' synset	Synset + hypo/hyper + SemCor	First lemma	**33**	26
2-4-2	'Best' synset	Synset + hypo/hyper + SemCor	Highest count	**33**	27
2-4-3	'Best' synset	Synset + hypo/hyper + SemCor	Most similar	**33**	21
2-5-1	'Best' synset	Synset + SemCor	First lemma	32	25
2-5-2	'Best' synset	Synset + SemCor	Highest count	32	26
2-5-3	'Best' synset	Synset + SemCor	Most similar	32	20

Table 1: different approaches (combinations of synset selection method, context depth, and lemma selection method) to the lexical substitution task, with corresponding accuracy scores. Methods are grouped by colour, and the best results are emboldened.

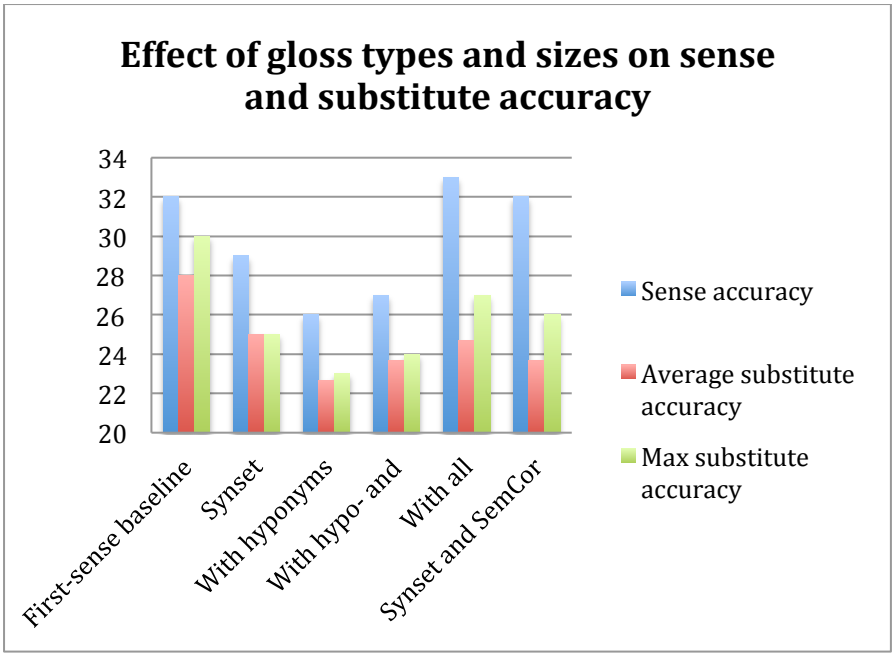

Chart 1: the impact of gloss types and sizes on the algorithm's ability to disambiguate word sense in context, and the corresponding impact on word substitution accuracy.

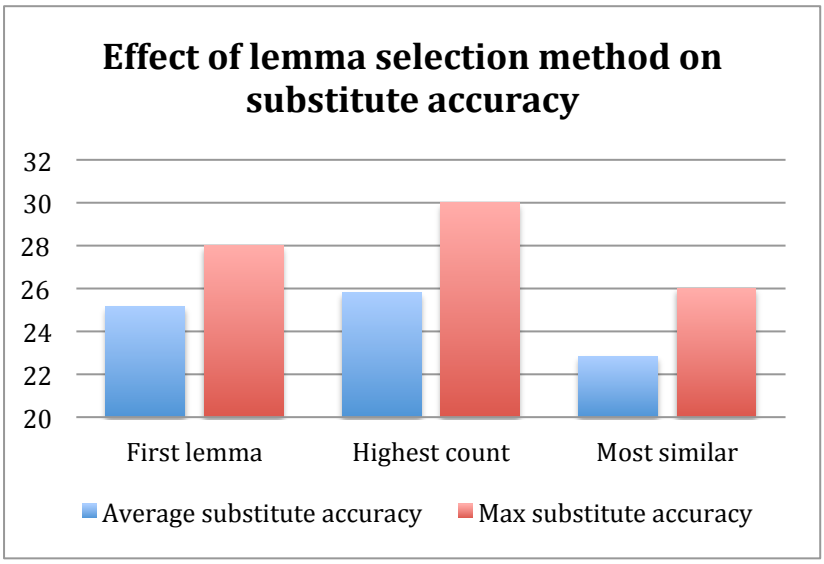

Chart 2: the impact of different approaches to lemma selection on the accuracy of selected substitutes, when compared against the human gold standard.

Analysis

As indicated by the presentation of two accuracy measures above, there are two broad areas in which an algorithm could go wrong on this lexical substitution task. The first relates to the more traditional WSD element of the problem: identifying the sense of a word that is being used in context. The second is with regard to the selection of an appropriate word representing the chosen sense.

At least for the knowledge-based approach outlined here, identifying the correct sense of a word is a fundamental pre-requisite for returning a suitable alternative word, and this problem is accordingly a sensible place to start. The tests above sought to establish to what extent the type and size of the glosses for the synsets considered affect the algorithm's ability to correctly identify the intended sense of a target word. It was expected that the larger the gloss, the greater the chance of meaningful crossover with the target context, and the better the sense disambiguation would be. However, the results presented in Chart 1 suggest a more complex relationship between the two.

What is surprising is that sense accuracy actually decreases using the first extensions of the simplified Lesk algorithm, where hypo- and/or hypernym definitions and examples are included in each synset gloss (compare the 'With hyponyms' and 'With hypo- and hypernyms' results from Chart 1 with the 'Synset' bars). This contrasts with the increase in accuracy seen when the bag-of-words features from SemCor sentences are included (the 'With all' results in Chart 1). On the surface, this would suggest that there is something inherently detrimental about the inclusion of hypo- and hypernym data. However, performance is better when all of the data is used than it is when SemCor data and the original synset gloss are used without consideration of hypo- and hypernyms (compare 'With all' to 'Synset and SemCor' in Chart 1).

This tentatively indicates a trade-off between the size and specificity of the gloss. When gloss sizes are small, crossover between them and the target context is likely to be extremely limited. Under these conditions, one word could swing the decision as to which sense the algorithm should choose. This, clearly, is not a strong foundation on which to make the decision, and opting for the most common synset is accordingly more successful (see the 'First-sense baseline' results in Chart 1). But the foundation becomes even weaker when hypo- and hypernym definitions and examples are added. These make the glosses less specific, so crossover words are less likely to be salient and the risk of the algorithm being drawn to an incorrect sense increases.

The SemCor data is arguably more specific, since it relates to the exact synsets and not related synsets. However, the improvement it offers seems more to do with it helping to overcome the issue of crossover sparsity. The extra improvement in performance seen when hypo- and hypernym information is added along with the SemCor data suggests that once the gloss size tips over a certain threshold, size of the gloss begins to matter more than specificity. In other words, when crossover is sparse it is better for it to be exact, while when crossover is expanded by a significantly larger gloss, any relevant additions are likely to improve performance.

Following this line of reasoning, the generally poor performance of the algorithm with regard to sense accuracy (struggling to better the first-sense baseline) can be explained by the fact that the glosses and the target contexts – and the resulting crossovers – are not sufficiently substantial for a best synset to be confidently identified. Expanding the glosses with information from other lexical resources would help, as would any increase in the target context size – perhaps even to document level, to make use of Gale's (1992) 'one sense per discourse' hypothesis. The use of supplementary lexical resources, if accompanied by applying an IDF measure to crossover words and ensuring repeated words are not counted twice, should also help overcome the current skew towards senses with bigger glosses.

It is not simply identifying the correct senses of target words that is important in this task, though. While sense accuracy beats that of the first-sense baseline when SemCor, hyponym, hypernym and original synset glosses are combined, the same does not happen for substitute accuracy. This is explicable in part by the fact that the first-sense baseline, in selecting common (and, accordingly, clearly defined) senses, tends also to end up with familiar words that, when the sense is correct, happen to match up with those selected by the human annotators. The same does not always happen for the less common senses selected by the best-sense approach. Regardless of approach, though, Chart 2 shows that selecting the lemma with the highest WordNet count (by calling its count() function) is more likely to return words in the gold standard than just taking the first lemma WordNet presents (some of which are quite unusual) or taking the most similar lemma to the original word, according to a distributional similarity measure (which is not helped by some lemmas not appearing in the vocabulary).

Even having identified a better lemma selection policy, substitute accuracy remains poor and markedly lower than sense accuracy across all methods. A number of reasons for this can be identified. First of all, the identification of alternative words is reliant on the lemmas provided with WordNet synsets. As noted above, some of these are unusual, and others are Americanised, resulting in many that would sound unnatural to a UK-based English speaker, including the human annotators on the task (the use of 'barroom', 'ginmill' and 'taproom' as opposed to 'pub', for instance). The failure to include some obvious words, such as 'crucifix' (which is astonishingly hard to get to from 'cross' using the WordNet interface) also complicates matters. The human annotations, too, can tend towards the bizarre: either of 'pn' or 'unknown person' would have been near impossible to get to as alternatives for 'gall', regardless of approach. As another example, the annotated substitutes for target words labelled as 'dark' are often replacements more specifically for a phrase - 'in the dark'. This, to a computer, would just seem against the rules.

Alternative Approaches

This lexical substitution task could be approached from a number of different angles, whilst staying within the limits of unsupervised systems. An alternative dictionary-based approach, for instance, could start by finding the synonyms of a target word – using, perhaps, the Lin thesaurus or a distributional similarity measure – before cross-referencing the synsets of these alternatives with the target context to identify a best substitute. Brief experimentation along these lines indicated that such an approach would be less sucessful than the approach above, but further investigation would be required to say for sure. Other dictionary-based approaches, such as finding the smallest semantic distance (Rada et al., 1989) or exploiting the graph-based nature of WordNet (Navigli and Lapata, 2010) would also be interesting to investigate.

An approach building on research into word sense induction might also be successful. Clustering methods drawing on those of Schütze (1998), with clustered context vectors used to denote the different senses of a word, could prove useful. It might even be possible to do away with the distinction between identifying a correct sense in the first instance, and alternative word in the second, by simply finding an alternative word that is used in a similar way. The wealth of unlabelled text data available online – through Wikipedia or Twitter, for instance – could be used to identify good substitute words based on those appearing in a similar context window to a given target, or those that appear in certain dependency relationships. These methods would have their own drawbacks (computational cost being one), but would likely return better alternatives than are retrieved when relying on WordNet.

Conclusion

Rather than outlining any ground-breaking new approach to WSD and lexical substitution, the above has served to demonstrate some of the limitations of a dictionary-based approach. Although a number of improvements to this implementation could be made – with the use of alternative lexical resources being perhaps the most obvious, further to the 'Analysis' section above – it is not clear that these changes would result in a substantial improvement in accuracy on the lexical substitution task. In light of the above, it is understandable that systems implementing Corpus Lesk tend to be used as little more than a baseline in such tasks (Jurafsky and Martin, 2017). Even in the absence of sense-labelled corpora, other unsupervised alternatives may prove more effective – and an approach incorporating bootstrapping would likely be better again, for not much more effort.

Bibliography

Banerjee, S. and Pedersen, T. (2003). Extended gloss over- laps as a measure of semantic relatedness. In *IJCAI 2003*, pp. 805–810.

Gale, W.A., Church, K.W., and Yarowsky, D. (1992). One Sense Per Discourse. In *DARPA Speech and Natural Language Workshop*, pp. 233-237.

Jurafsky, D. and Martin, J. H. (2017). Computing with Word Senses: WSD and WordNet. In *Speech and Language Processing* (3rd edition draft). Available here: https://web.stanford.edu/~jurafsky/slp3/ [Accessed 11/03/17]

Lesk, M. E. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Inter- national Conference on Systems Documentation*, Toronto, CA, pp. 24–26.

McCarthy, D. and Navigli, R. (2007). Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations.*

Navigli, R. and Lapata, M. (2010). An experimental study of graph connectivity for unsupervised word sense disam- biguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(4), pp. 678–692.

Rada,R., Mili, H., Bicknell, E. and Blettner, M. (1989) Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 19, pp. 17-30.

Schutze, H. (1998) Automatic Word Sense Discrimination. *Computational Linguistics* 24(1), pp. 27-93.

Vasilescu, F., Langlais, P., and Lapalme, G. (2004). Evaluating variants of the lesk approach for disambiguating words. In *LREC-04*, Lisbon, Portugal, pp. 633–636. ELRA.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *ACL-95*, Cambridge, MA, pp. 189–196.