Unsupervised Learning of Syntactic Similarity

Simon Fung
University of Alberta
Edmonton, AB
simonmin@ualberta.ca

1 Introduction

This paper describes a computational method, or algorithm, that learns the degrees of similarity in *lexical-syntactic behaviour* (syntactic behaviour that is lexically determined) among words in a corpus. This is similar to learning parts of speech, except that the words are not grouped into discrete categories; rather, each is represented by its degree of similarity to all other words in a corpus. This graded, polythetic model captures both the differences within each traditional part of speech and the similarities across different parts of speech.

The algorithm is *unsupervised*, meaning it learns with no answers given – the same way humans learn. As such, besides its practical uses, it can also be thought of as a model for the acquisition of syntactic behaviour.

In its current state, the algorithm is rather computationally-expensive, and can only been tested on small corpora. Nevertheless, results are encouraging. Work is ongoing, and the modification of several aspects of the algorithm, together with the use of larger corpora, should lead to performance improvements.

1.1 Motivations

The work described in this paper is motivated both by scientific inquiry and practical application, by the desire to better understand the nature of lexical-syntactic behaviour with a computational model, and by the desire to provide other natural language processing applications with infomration about lexical-syntactic behaviour.

Lexical-syntactic behaviour, most often described in terms of parts of speech, has traditionally been studied by discerning patterns based on a handful of words with typical behaviour. A major advantage of performing a computational analysis on corpora, compared to traditional linguistic analysis techniques, is that it can analyze the syntactic behaviour of all words found in the training data. In language, there can often exceptions to the most obvious patterns, and when these exceptions are many, they can provide an overall view of a phenomenon like lexical-syntactic behaviour that is fundamentally different from one arrived at by focusing only on typical cases.

While lexical-syntactic behaviour is a familiar subject of linguistic inquiry, it also has practical applications. By identifying which words have similar syntactic behaviour, the algorithm can enrich language-specific information for a variety of natural language applications, including machine translation and parsing. Information about each word will no longer be limited to its occurences in the corpora, but will also include occurences of syntactically similar words.

In the next section I will argue for excluding the semantics of words when characterizing their lexical-syntactic behaviour, something that a computational model naturally does. Given that definition, I will also argue against a categorical view of lexical-syntactic behaviour based on equivalence classes, in favour of a more graded, polythetic view, based on the notion of family-resemblance (Wittgenstein 1953), a position that informs the algorithm's design.

2 Lexical-Syntactic Behaviour

Lexical-syntactic behaviour, which determines a word's part of speech, refers to the range of morphosyntactic contexts a word can occur in. A word's morphosyntactic context consists of both the words and phrases occurring around it, and the clitics and affixes bound to it. For example, from a sentence such as *I went to two parties last night*, the context of *party* would be *I went to two* ____*s last night*; the size of the relevant context

can differ, depending on the constructions involved.[1]

The semantics of a word are sometimes considered to be the determinants of its part of speech; however, if parts of speech are to be linguistic phenomena, a property of languages, then this cannot be so. Semantics are real-world concepts that exist outside of language, so a classification of words based solely on their semantic content would just be a classification of real-world concepts, and would not be a property of languages at all. In fact, the similarities and differences in the morphosyntactic behaviour of words in a language already reflect a particular way of organizing real-word concepts. Often, two concepts are expressed by words that occur in similar morphosyntactic contexts in one language, but in undeniably different ones in another. For example, while in English *red* and *green* both occur in similar morphosyntactic contexts, the corresponding Japanese words, *akai* and *midori*, do not: *midori* behaves like a Japanese noun, while *akai* behaves like a Japanese adjective.

It is interesting that there are consistent patterns in the way languages organize real-world concepts. As Beck (2002:41) argues, the relationship between the meaning and the syntactic behaviour of words is not arbitrary. Cross-linguistically, among lexical classes, certain concepts tend to share morphosytactic contexts. Some concepts are "conceptually autonomous" ("[existing] independently of the events in which they are participants"), "temporally stable," and serve as "semantic names"; these are typically referred to as nouns. Other concepts are conceptually "non-autonomous," "temporally unstable," and serve as semantic "predicates"; these are typically referred to as verbs. Other concepts, however, partially resemble each prototype but that do not fit in either one completely, and it is these that show the greatest cross-linguistic variation in their morphosyntactic behaviour: some behave like nouns in their language, others behave like verbs, and still others behave differently from either type. In particular, property concepts (such as 'red' and 'green') and human characteristics, which share some semantic properties of nouns and of verbs as typical semantic names and predicates, expectedly show much cross-linguistic variation.

This relationship between syntactic behaviour and semantics is a fascinating reflection through language of how humans mentally organize the world, and is probably why the semantics of words have been used to characterize parts of speech. However, even if this relationship were empirically observed to be perfectly consistent cross-linguistically, theoretically it is still not necessarily so. Even if the words for 'red' and 'green' had the same morphosyntactic behaviour in every language, nothing would exclude the possibility of the discovery of a new language where they behaved differently. Defining parts of speech based on semantics, however, would ignore any such differences, and would therefore be useless in describing language. Therefore, parts of speech should be defined by morphosyntactic behaviour. There appears to be semantic correlations to morphosyntactic behaviour, but they are far from perfect, and there need not be any.

Parts of speech, then, are categories of words that exhibit the same morphosyntactic behaviour. Conventionally, they have been viewed as equivalence classes: each part of speech has a set of membership requirements (often referred to as "tests"), which, once fulfilled, guarantees membership. These are also known as hard or discrete categories: either a word is in one, or it is not.

Theoretically, words in some language could align themselves very clearly into groups with uniform morphosyntactic behaviour, which share no contexts with any other. In that case, grouping them into discrete parts of speech would not result in the loss of behavioural detail. However, this appears not to be the case with natural human languages. This has been noted both in linguistics as early as 1924 by the Danish linguist Otto Jespersen (Jespersen 1924), as well as in the computational linguistics community, where Schuetze (1995) notes that "it is by no means generally accepted that such a [hard] classification is linguistically adequate." In practice, there is often variation within conventional parts of speech, perhaps out of a reluctance to create too many small categories. Subcategories are sometimes posited, though even those are sometimes not fine enough to distinguish all nuances, especially words whose usage is limited to certain idiomatic expressions (e.g. *aback*, though labeled an adverb[2], is almost never used except in the

[1] Affixes may, of course, differ in their degree of analyzability.

[2] in the Merriam-Webster dictionary (http://www.merriam-webster.com/dictionary/aback).

expression *to be taken aback*[3]). Moreover, hard categories not only fail to reveal differences within each category, but also similarities among different ones. Adjectives and nouns in English are different in some respects: for example, adjectives cannot be pluralized, and cannot occur by themselves (as mass nouns can) or take the indefinite article, without a noun to modify (as count nouns can). However, they also share similarities: adjectives can take *the* and behave like definite nouns (*the rich, the poor*), precede and modify nouns like modifying nouns in a noun compound (*big trucks* vs. *monster trucks*), and form predicates with *to be* (*this is hard* vs. *this is water*). Thus, the generalization of words into hard categories is a simplification that ignores their details.

On the other hand, a family-resemblance (Wittgenstein 1953) or polythetic view of lexical-syntactic behaviour acknowledges that words can resemble each other in morphosyntactic behaviour, but also differ in various, sometimes subtle ways. In this view, words can resemble each other to differing degrees. They can be split into groups for various purposes, if certain details can be ignored; however, these groupings would occur after the different degrees of similarities among words are recognized. Words would be viewed as points spread out over some space, like corn kernels spilled onto a floor.

The two motivations described in §1.1, together with the theoretical perspectives argued for in this section, have driven some of the design decisions that distinguish the model in this paper from previous efforts. To highlight these differences, some notable previous efforts will be described in the next section.

3 Previous work

The major previous efforts to learn parts of speech share many similarities, and are best viewed as variations on two approaches, which I will refer to as the *context-vector approach*, and the *function-maximizing approach*.

The context-vector approach can be broken down into several steps. First, the context of a word is defined. Finch & Chater (1992) defines a context of a word as the preceding two and following two words, as does Schütze (1995), Freitag (2004), and Biemann (2006); Gamallo et al. (2004) uses only one word on each side.

Next, as in Finch & Chater (1992) and Schütze (1995), the algorithm counts how many times context words occur in each neighbouring position for each word. For example, it would record how many times *the* and *and* appear to the left of *dog* respectively, and so on. It would repeat this for all context words, then for all positions in the context, then for all words in the corpus. Because of the rarity of many words in the corpus, the context words are usually limited to the most frequent words in the corpus: Finch & Chater (1992) uses the most frequent 150, while Biemann (2006) uses the most frequent 200, and Schütze (1995) uses the most frequent 250. Rather than taking the most frequent words as features, Gamallo et al. (2004) first finds functional words by extracting words that are common to two texts on different subjects. In all cases, the result is a vector of context-word frequencies for each word in the corpus (see Figure 1).

Finally, the words are compared and classified based on the correlation of their context-word frequency vectors. Either they are grouped into a pre-specified number of categories, as in Schütze (1995) and Freitag (2004), or they are grouped together if some similarity threshold is satisfied, as in Finch & Chater (1992) and Biemann (2006). Schütze (1995) reduces context vectors to core independent dimensions, using principle component analysis, before clustering. In another variation, Biemann (2006) clusters graphs representing the 10,000 highest frequency words and the words lower in frequency than the 2000th most frequent word separately, then merges the information from the two partially overlapping graphs.

One aspect of this context-vector approach that simplifies linguistic reality is in keeping track of context words individually, and not together. Doing this means not keeping track of what words in other positions were present with each occurrence of that context word. In the string *the dog is*, for example, the occurence of *the* on the left would be completely divorced from the occurence of *is* on the right. Yet such information is important in distinguishing a word like *dog*, which can occur between *the* and *is*, and another word like *always*, which can occur after *the* (as in *the always lovely Natalie Portman*), and before *is* (as in *it always is*), but not usually both at once (**the always is*).

Another approximation made in past algorithms is to represent words as points in a multi-dimensional Euclidean space, with contexts as

[3] From *Corpus of Contemporary American English.*

orthogonal dimensions. First, it is inaccurate to represent all contexts as mutually orthogonal dimensions; the context words that provide clues about a target word's similarity to other words are at some point target words themselves, and whose varying degrees of similarity to other words we are seeking to measure. Thus, they cannot be suddenly assumed to be independent from each other when used as context words.

These correlated dimensions could theoretically be reduced to a smaller number of independent dimensions, using some form of principal component analysis. However, if we want to use left-right context pairs, as stated above, there will be $|V|^2$ left-right context pairs (where $|V|$ is the size of the vocabulary). This means that each word will be represented by n^2-dimensional vectors, before the dimensions can be reduced. For a small corpus with a vocabulary size of 100,000, this would produce vectors with 10 billion dimensions, a rather infeasible number.

Finally, algorithms that only use the most frequent words as context words lose much contextual information. The motivation for using a small subset is sparseness: context words that occur with very low frequency do not say very much about the syntactic behaviour of the words they are contexts of – that is, unless we realize that these rare words are not independent contexts, but rather have similar syntactic behaviour as many other words in the text. As the algorithm measures the similarities among words in the corpus, it is also measuring the similarities of context words, since context words and the words it is trying to classify are the same words. With more words being identified as being similar to these rare context words, the sparseness problem can be overcome. This approach makes use of more contexts.

In the function-maximizing approach, a function that correlates with accuracy is first defined (e.g. the likelihood of the data, the likelihood of the part-of-speech tags, mutual information between words and tags). Then, the algorithm searches for the word-tag correspondence that maximizes the function. With the number of classes pre-specified, Brown et al. (1992) puts each of that same number of the most frequent words in its own category, then puts each of the other words in the category that would increase the likelihood of the data. Similarly, starting with some initial clustering, Clark (2003) moves words one at a time to a different category, to find a classification that maximizes the corpus's probability. It is also supplemented by a letter-based probability model that encourages words with similar letter distributions to group together. Freitag (2004) does something similar for high-frequency terms, finding categories both of words to be classified and the contexts they occur in by moving a random word or context to a random new category, in order to maximize the mutual information (roughly the predictability) between categories of words and their contexts. However, as Freitag (2004) notes, this approach does not work as well for low-frequency words as it does for high-frequency ones, due to data sparseness. For low-frequency terms, it uses a Hidden-Markov Model (HMM) to maximize a likelihood function.

Hidden-Markov Models were used in earlier algorithms for part-of-speech disambiguation, a task which resembles part-of-speech induction, but where each word is constrained to the parts of speech that it has been observed to belong to in the training corpus. More recently, however, they have been used in unconstrained part-of-speech induction as well (Goldwater and Griffiths 2007, Johnson 2007, Graca et al. 2009, Berg-Kirkpatrick et al. 2011, Blunsom and Cohn 2011). In a first-order Hidden-Markov Model, part-of-speech tags are modeled as hidden states to be inferred, each with its own parameters: *emission probabilities* for emitting various words, and its own *transition probabilities* to other states (citation?). For linear sequences of words, which are provided by the training corpus, various algorithms are used to find both the state or tag sequence and the parameters most likely to have produced the observed word sequence. Moreover, since in testing a tagger will encounter words not present in training data, probability is often shifted from observed words to unobserved ones; techniques for this procedure, known as *smoothing*, also varies. Blunsom and Cohn (2011) uses more sophisticated methods of smoothing probabilities of unseen words and tag sequences, resulting in the current state-of-the-art performance for unsupervised part-of-speech induction.

Such function-maximizing algorithms avoid some of the inaccuracies of those of the first type; however, there is still room for improvement. By categorizing words before using them in contexts for other words, information is lost. More fundamentally, maximizing a likelihood function does not necessarily lead to higher accuracy (Merialdo 1994, Liang and Klein 2008).

Although more recent methods have employed increasingly sophisticated statistical techniques,

their performance has not markedly improved on earlier, simpler efforts. Christodoulopoulos et al. (2010) evaluated seven algorithms, spanning 20 years, and finds that the older ones, such as Brown et al. (1992) and Clark (2003), perform better than some of the newer, more sophisticated efforts. In fact, Clark (2003) performs the best overall, with its only rival being Berg-Kirkpatrick et al. (2010), the most recent of the seven systems.

Schütze and Walsh (2008) stands out among the numerous efforts using discrete categories, with their graded, non-categorical model. They represent each word with a four probability distribution over all observed words: one for the probability of occuring to the left of the target word, and one for occuring to the right; and one for the probability of occuring in place of the target word based on the left context, and one based on the right context. In training, sequences of these probability distributions are stored in memory; then, during evaluation, sequences are judged for grammaticality based on whether or not a sequence is sufficiently similar to a stored sequence. This model is inspired by the same problems of categorical models as the algorithm in this paper. However, unlike the algorithm in this paper, Schütze and Walsh (2008) assumes independence both between left and right contexts, as well as among words in each probability distribution.

These observations have inspired specific design decisions in the algorithm in this paper. The next section will describe the details of the algorithm itself, and will discuss these design decisions in the process.

4 The algorithm

The algorithm can be described in two parts: first gathering the contexts of each word in the corpus, then calculating the similarities of each pair of words based on their contexts.

4.1 Gathering contexts

As input, the algorithm takes text that has been segmented into sentences, lower-cased, and stripped of punctuation (excluding some hyphens, apostrophes, and periods for abbreviations). Then, for each word token in the text, the contexts it occurs in are recorded and stored. A context of a word is taken to be the immediately preceding word plus the immediately following word. For example, in the string *the dog is*, the context of this occurence of *dog* consists of *the* and *is*, and can be represented as [‘the’, ‘is’]. The symbol ‘^’ is used to mark the beginning of a sentence, and ‘$’ is used to mark the end of a sentence; however, these symbols have no recorded contexts themselves. The result is a mapping between each word and the set of contexts that it appears in in the text; I will refer to this as the *word-context map*.

The algorithm does not keep track of the number of times the contexts of each word occur with that word. Beck (2009:22) argues that frequency counts are an unreliable indicator of markedness; a similar argument can be made against using frequencies to count more frequent contexts more heavily. The frequency of a specific context of a word often depends on factors unrelated to the grammar of the language, such as pragmatics and subject matter. On the other hand, the mere presence of a context for a word, as opposed to its absence, is evidence that it is in the realm of possibility, and is much more informative than its frequency.

4.2 Calculating similarity scores

The calculation of similarity scores is the main part of the algorithm. The similarity score between every combination of two word types is taken to be the similarity between their sets of contexts.

With conventional mathematical sets, the intersection between two sets is the set of items that are appear in both sets. In the sets A = {Alice} and B = {Alice, Bob}, only Alice is common to both sets; Bob is considered a completely person from Alice.
However, consider these two sets of contexts:

A = {[‘the’, ‘is’]}
B = {[‘the’, ‘is’], [‘a’, ‘ran’]}

Since ‘the’ and ‘a’ may be similar in syntactic behaviour, as may ‘is’ and ‘ran’, the contexts [‘the’, ‘is’] and [‘a’, ran’] can have a similarity score anywhere from 0 to 1. Therefore, [‘a’, ran’] can be thought of as being partially belonging to set A, having some of its information shared by [‘the’, ‘is’]. The degree of similarity between these two contexts, then, determines the degree of similarity between sets A and B.

The definition of set intersection used in this algorithm is inspired by fuzzy set logic. In fuzzy set logic, an item has a graded degree of membership to a set, anywhere from 0 to 1. An item’s membership to the intersection set of two sets is taken to be the smaller of its degrees of member-

ship to the two sets (note that the intersection is itself a fuzzy set).

In this algorithm, when comparing the context sets of two word types, each context belongs fully to the context set of the word that it is in fact a context of, but can also belong partially to the other set. Its partial membership to the other set is defined to be its similarity score with the most similar context in this other set. In this simple example, set A only has one context, and so ['a', 'ran'] only has one context to compare to; if set A had more contexts, the membership of ['a', 'ran'] to set A would be its similarity score with the most similar context to it in set B. So if ['the', 'is'] and ['a', 'ran'] have a similarity score of 0.5, the membership of ['a', 'ran'] to set A would be 0.5. Since the membership of ['a', 'ran'] to set B is 1, its membership to the intersection set would be 0.5, the smaller of the two memberships. Since ['the', 'is'] belongs fully to both sets A and B, it has a membership of 1 to the intersection set. The size of the intersection set, then, is 1.5. The similarity score of the two words with A and B as their respective context sets would be the size of the intersection set divided by the size of their union, which is 1.5/3 = 0.5. If set A contained other contexts, the membership of ['a', 'ran'] to set A would be the average of its similarity score to all the contexts in set A.

The similarity between two contexts is taken to be the average similarities of the two words in each context position, across all context positions. For example, the similarity between ['the', 'is'] and ['a', 'ran'] is the average of the similarity score between 'the' and 'a' and the similarity score between 'is' and 'ran'. The similarity scores between two words are obtained from a *similarity table*, which stores the similarity scores between any two words appearing in the corpus.

Initially, there is no similarity table, so two context words either have a similarity score of 1 if they are identical, or 0 otherwise. Every pair of context sets is compared, and the scores for each pair fills in the similarity table. The pairs of context sets are then compared again, this time using the scores from the similarity table. This is repeated until the average difference in similarity score between the newest similarity table and the last one falls below a certain threshold. The final output is the similarity table from the final iteration, with similarity scores between every pair of words in the training corpus.

4.3 Training

The algorithm was trained on English text from Penn Treebank-3. Because of the slowness of the algorithm, only small subsections of it were used: it was first trained on the first 46 sentences (1,008 words) of the treebank, and then on the first 500 sentences (10,201 words). Although these are much smaller than the million-plus words most current algorithms are trained on, a significant improvement in performance with a corpus ten times larger (the 10,201 words) would suggest potential improvements with even larger corpora sizes.

5 Evaluation

Evaluating automatically-learned clusters of words is not straightforward (Christodoulopoulos et al. 2010). The standard they are evaluated against is usually part-of-speech tags in annotated corpora; however, automatically-learned clusters are not assigned any tags by the algorithm, raising the question of how each cluster should be labeled. Moreover, the number of learned clusters sometimes differs from the number of parts of speech in the standard, raising the question of whether or not to let more than one cluster map to each part of speech in the standard. Evaluating automatically-learned similarity scores is even more problematic: there is the question of whether or not to first cluster words based on these similarity scores.

Since one of the main goals of this algorithm is to try to capture diversity in lexical-syntactic behaviour by not clustering, its output would preferably be evaluated in its original non-clustered form. One way to evaluate the similarity scores without clustering is to rank, for each word type (the *reference* word type), all other word types in the corpus (the reference's *neighbours*) by their similarity scores with the reference, according to the algorithm. These rankings can then be evaluated by comparing the parts of speech of the reference and each neighbour in its ranking, as given by a corpus tagged with parts of speech. The best ranking would be one where neighbours that share at least one part of speech with the reference are all ranked higher than those that do not. This would require normal part-of-speech tags, which are discrete and thus would not reveal differences within each category (e.g. singular count nouns and mass nouns both share a tag), nor similarities between categories (e.g. between pronouns and nouns). They may also take into factors beyond syntax (e.g. differentiat-

ing between regular, comparative, and superlative adjectives, which syntactically behave very similarly). Nevertheless, a tagged corpus is the most conveniently accessible standard for lexical-syntactic behaviour of words in a large body of text. Therefore, it will be used here for evaluation purposes, with those caveats in mind.

For this evaluation method, there is no need to separate training and testing data; it is not a task that indirectly tests the accuracy of the model, but a direct evaluation of the model (i.e. similarity rankings, based on similarity scores) itself. Introducing fresh data would affect all the similarity scores among words, and would thus have the same effect as adding training data. The purpose of using fresh testing data is to test the model resulting from training; evaluating rankings for each word would do this directly.

The metric I use to evaluate these rankings is mean average precision, a technique used in information retrieval to evaluate the ranking of returned documents for a search query. In the ranking of the neighbours of each reference word type, at each rank *k* that is occupied by a neighbour sharing at least one part of speech tag with the reference (a *matching neighbour*), the total number of matching neighbours in the top *k* documents is divided by *k*, giving a precision score for the top *k* documents. This results in the same number of precision scores as there are matching neighbours in the ranking. In the ideal case, where all neighbours that are matching are ranked above those that are not, the precision would be 1.0. The average precision, then, is the average of all these precision scores, and the mean average precision is the average of these average precisions over all reference word types.

The baseline average precision is random ranking: for a reference word type, given the number of matching neighbours and the number of words in the ranking, it is the expected average precision over all possible orderings of matching and non-matching neighbours, with each ordering being equally probable.

A general summary of the results is given in Table 1:

Table 1: Maxium, minimum, and mean average precisions for subsets of Penn Treebank-3.

	Mean average precisions (%)		Average precisions (algorithm) (%)		Ratios of algorithm-to-baseline average precisions (%)		
	Alg.	Base.	Max.	Min.	Max.	Min.	Avg.
A	17.7	11.5	60.6	0.223	23.1	0.155	1.93
B	20.1	13.8	63.3	0.0442	117.8	0.156	2.20

A = First 1,008 words of Penn Treebank-3
B = First 10,201 words of Penn Treebank-3

The algorithm's mean average precision rates are rather low. Nevertheless, its improvements on the baseline precisions are significant, implying that some of what the algorithm has gotten correct is not due to chance. Ratios of the algorithm's average precision to the baseline's average precision were computed for each word type; the minimum, maximum, and average of these ratios are given in Table 1. The ratios are more indicative of performance than average precision rates alone, since they take the baseline into account. The average ratio over all word types is higher for the larger corpus than for the smaller one (1.93 vs. 2.20); the maximum ratio in the larger corpus is also higher than the maximum ratio in the smaller corpus, while the minimum ratios are about the same. These numbers suggest that training the algorithm on even larger corpora should further improve its performance.

6 Discussion

Some of the algorithm's errors seem to be due to the parts of speech used in the Penn Treebank. Some parts of speech found in the corpus seem to be nearly syntactically identical or at least quite similar, and differ mostly in ways beyond syntax. Such is the case with proper nouns and personal pronouns. Their differences, which are syntactically very slight, were largely not identified by the algorithm, and may have caused average precision rates to be lower than they would be otherwise.

Several flaws in the algorithm itself could also be observed in the results. The predominant source of error appears to be due to the insufficiency of using contexts consisting only of one neighbouring word to the immediate left and right. For example, ['to', 'and'] is a context for verbs such as 'develop' and 'mature,' but also for nouns like 'construction' and 'securities.' However, when a wider context is considered, the possibilities for the target word becomes more limited. For example, a wider context for

the occurence of ‘develop’ is “. . . just before the pollen is about to ____ and it . . . ,” the underscore showing the position of ‘develop.’ In this wider context, it is apparent that ‘develop’ behaves like a verb. The contexts currently used by the algorithm are too small to carry all the contextual information available for predicting their target words.

Another problem is that the algorithm attributes too much importance to shared contexts that are actually not very informative, contexts that occur with words with wide-ranging syntactic behaviours. The algorithm gives each context word equal weight, and thus fails to account for differences in informativeness. For example, ‘^’ and ‘$’, the markers for the beginning and the end of a sentence respectively (added in post-processing, and are not actual words in the corpus), are part of the contexts of many words with diverse syntactic behaviours. By giving all contexts equal weight, the algorithm overestimates the contribution of less informative contexts, while underestimating the sharing of more informative ones.

Still others constrain the target word with both the left and right context words working together. An example is the context [‘each’, ‘of’], the only context for the word ‘type.’ Both ‘each’ and ‘of’ work together to constrain the word between them to be a singular count noun. Individually, however, they are less selective. The word ‘purged’ has [‘is’, of’] as its only context. With only ‘of’ in common with [‘each’, ‘of’], [‘is’, ‘of’] can no longer be relied upon to select for singular count nouns only. Since the algorithm has also mistakenly judged ‘is’ to be quite similar to ‘each,’ the word ‘purged’ is consequently judged to be quite similar to ‘type.’

Some of these less informative contexts occur at phrasal boundaries, which are no longer marked after the removal of punctuation. Consider this sentence: “Stuffing a wad of Red Man into his cheek he admits the fastball he brought into the majors in 1955 has become a slowball.” ‘He’ is considered by the algorithm to be the right context of ‘cheek’; however, since there is a phrasal boundary between ‘cheek’ and ‘he’, the right context can be one of many words with very different behaviours, and so is not very informative. The removal of punctuation was originally meant to approximate spoken speech; however, speech has prosodic information that may signal phrase boundaries, which text does not benefit from. Future evaluations may retain punctuation marks.

Finally, many of the errors appear to be due to low frequencies, not surprising given the small sizes of the corpora. When the word types in each corpus are ranked by their ratio of average precision of the algorithm to that of the baseline, the top ranking words are dominated by high-frequency function words: the top ten words from the 10,201-word corpus are ‘which,’ ‘can,’ ‘might,’ ‘would,’ ‘should,’ ‘their,’ ‘or,’ ‘his,’ ‘biggest,’ and ‘could.’ The words with the worst performances are more mixed, but include many multi-word proper nouns, as well as low-frequency function words; puzzlingly, the word with the worst performance is ‘who,’ which has a relatively high frequency in the corpus, and which is very similar to ‘which,’ the word with the best performance. Occasionally, two adjacent words with single occurences can be found that occur in the same context. For example, in the phrases “from Dave Kingman” and “from fertilizing itself,” both of which occur at the end of their respective sentences, the algorithm considered ‘Dave’ to be syntactically-identical to ‘fertilizing,’ and ‘Kingman’ syntactically-identical to ‘itself,’ since all four words are single occurences in the corpus.

In summary, while the evaluation process can be modified to more accurately reflect the algorithm’s performance, there are areas to improve in the algorithm itself.

7 Future Work

Although the results contain a fair amount of noise, they are nevertheless encouraging. Most significantly, the error analysis does not reveal factors that contradict the theoretical assertions underlying the algorithm’s basic model.

The sources of errors, as discussed in the last section, point to at least three areas for improvements. Firstly, the algorithm’s definition of context has shown to be too narrow, and should be expanded to include all words that influence the prediction of the target word. The informativeness of context words should also be considered. These contexts, which will be of differing sizes and can be discontinuous, must then be compared in a way that is more linguistically-informed than simply averaging similarity scores between context words of two contexts in the same position, as this algorithm currently does.

Secondly, the algorithm currently ignores morphology, which provides substantial information about a word’s syntactic behaviour in many languages. The two top-performing systems evalu-

ated by Christodoulopoulos et al. (2011) both take morphology into account when inducing parts of speech (Clark 2003, Blunsom and Cohn 2011), suggesting that morphology is indeed important to consider. It will be incorporated into future versions of the algorithm.

Finally, on the technical side, the algorithm currently runs too slowly for it to be trained on larger corpora. The quantity of training data alone is often the most important factor in an algorithm's performance (Brill 2003). This issue, then, must be addressed, in order to observe the true potential of the algorithm.

Beyond automatically learning the syntactic similarities of words, the larger goal is to use this information to automatically identify and learn the syntactic similarities of phrases, to produce a sort of syntactic analysis. Currently, most parsing algorithms use part-of-speech tags as input. Using graded, non-categorical lexical-syntactic information instead of part-of-speech tags may improve parsing performance by providing more accurate information; Schütze and Walsh (2008) shows that in a grammaticality judgement task, their non-categorical model significantly outperforms the categorical model by Redington et al. (1998), for sequences of three words or longer. Schütze and Walsh (2008) also provides an intuitive way to identify and measure similarities among larger phrases, simply by comparing the syntactic behaviour of words in the phrases, position-by-position.

In the other direction, phrasal syntactic similarities can also benefit the learning of lexical-syntactic information: more complete contexts, and information about degrees of similarity among them, can improve similarity scores among words, which in turn would improve the similarity scores among phrases, and so on, in a virtuous cycle. The result of all this will hopefully be a more nuanced syntactic analysis than what phrase structures and syntactic dependencies allow, by uncovering the degrees of similarities within and across different types of phrasal elements.

References

Beck, D. (2002). *The typology of parts of speech systems: The markedness of adjectives*. New York: Routledge.

Beck, D. (2009). A taxonomy and typology of Lushootseed valency-increasing suffixes. *International Journal of American Linguistics* 75, 533–569.

Berg-Kirkpatrick, T., Côté, A. B., DeNero, J., Klein, M. (2011). *Painless unsupervised learning with features.* In *Proceedings of NAACL 2010*, 582-590. Los Angeles, California.

Biemann, C. (2006). Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of COLING ACL 2006*, 7-12. Morristown, NJ.

Blunsom, P., Cohn, T. (2011). A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction. In *Proceedings of ACL 2011*, 865-874. Portland, Oregon.

Brill, E. (2003). Processing Natural Language without Natural Language Processing. In *Proceeedings of the 4th International Conference on Computational Linguistics and Intelligent Text Processing and Computational Linguistics (CICLing 2003)*. Mexico City, Mexico.

Brown, P. F., Della Pietra, V. J., Desouza, P. V., Lai, J. C., Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational Linguistics,* 18(4), 467-479.

Christodoulopoulos, C., Goldwater, S., Steedman, M. (2010). Two Decades of Unsupervised POS induction: How far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Clark, A. (2003). Combining distributional and morphological information for part of speech induction. In *Proceedings of EACL 2003*, 59–66. Morristown, NJ.

Finch, S., Chater, N. (1992). Bootstrapping syntactic categories using statistical methods. In *Background and Experiments in Machine Learning of Natural Language: Proceedings of the 1st SHOE Workshop on Statistical Methods in Natural Language*, 229-235. Katholieke Universiteit, Brabant, The Netherlands.

Freitag, D. (2004). Towards Unsupervised Whole-Corpus Tagging. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, 357-363. Geneva, Switzerland, Aug 23-27, 2004.

Gamallo, P., Lopes, G. P., Da Silva, J. F. (2004). A divide-and-conquer approach to acquire syntactic categories. In *Grammatical Inference: Algorithms and Applications, 7th International Colloquium (ICGI 2004)*, 151-162. Athens, Greece, October 11-13, 2004.

Goldwater, S., Griffiths, T. (2007). A fully bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of ACL 2007*, 744–751. Prague, Czech Republic.

Graca, J., Kuzman, G., Taskar, B., Pereira, F. (2009). Posterior vs parameter sparsity in latent variable models. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, (eds.), *Advances in Neural Information Processing Systems* 22, 664–672.

Jespersen, O. (1924). *The Philosophy of Grammar*. Chicago: University of Chicago Press.

Johnson, M. (2007). Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, 296–305. Prague, Czech Republic.

Liang, P., Klein, D. (2008). Analyzing the Errors of Unsupervised Induction. In *Proceedings of ACL 2008*, 879-887. Columbus, Ohio, June 16-18, 2008.

Merialdo, B. (1994). Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2), 155-172.

Schütze, H. (1995). Distributional part-of-speech tagging. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL).*

Schütze, H., Ward, M. (2008). A graphical-theoretic model of lexical syntactic acquisition. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008).* Honolulu, Hawaii, October 25-27, 2008.

Wittgenstein, L. (1953). *Philosophical Investigations*. Englewood Cliffs: Prentice-Hall.