Automated Multiword Expression Prediction for Grammar Engineering

Yi Zhang & **Valia Kordoni**
Dept. of Computational Linguistics
Saarland University
D-66041 Saarbrücken, Germany
{yzhang,kordoni}@coli.uni-sb.de

Aline Villavicencio & **Marco Idiart**
Institutes of Informatics & Physics
Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500
Porto Alegre - RS, Brazil
avillavicencio@inf.ufrgs.br
idiart@if.ufrgs.br

Abstract

However large a hand-crafted wide-coverage grammar is, there are always going to be words and constructions that are not included in it and are going to cause parse failure. Due to their heterogeneous and flexible nature, Multiword Expressions (MWEs) provide an endless source of parse failures. As the number of such expressions in a speaker's lexicon is equiparable to the number of single word units (Jackendoff, 1997), one major challenge for robust natural language processing systems is to be able to deal with MWEs. In this paper we propose to semi-automatically detect MWE candidates in texts using some error mining techniques and validating them using a combination of the World Wide Web as a corpus and some statistical measures. For the remaining candidates possible lexico-syntactic types are predicted, and they are subsequently added to the grammar as new lexical entries. This approach provides a significant increase in the coverage of these expressions.

1 Introduction

Hand-crafted large-scale grammars like the English Resource Grammar (Flickinger, 2000), the Pargram grammars (Butt et al., 1999) and the Dutch Alpino Grammar (Bouma et al., 2001) are extremely valuable resources that have been used in many NLP applications. However, due to the open-ended and dynamic nature of languages, and the difficulties of grammar engineering, such grammars are likely to contain errors and be incomplete. An error can be roughly classified as *under-generating* (if it prevents a grammatical sentence to be generated/parsed) or *over-generating* (if it allows an ungrammatical sentence to be generated/parsed). In the context of wide-coverage parsing, we focus on the *under-generating* errors which normally lead to parsing failure.

Traditionally, the errors of the grammar are to be detected manually by the grammar developers. This is usually done by running the grammar over a carefully designed test suite and inspecting the outputs. This procedure becomes less reliable as the grammar gets larger, and is especially difficult when the grammar is developed in a distributed manner. Baldwin et al. (2004), among many others, for instance, have investigated the main causes of parse failure, parsing a random sample of 20,000 strings from the written component of the British National Corpus (henceforward BNC) using the English Resource Grammar (Flickinger, 2000), a broad-coverage precision HPSG grammar for English. They have found that the large majority of failures are caused by missing lexical entries, with 40% of the cases, and missing constructions, with 39%.

To this effect, as mentioned above, in recent years, some approaches have been developed in order to (semi)automatically detect and/or repair the errors in linguistic grammars. van Noord (2004), for instance, takes a statistical approach towards semi-automated error detection using the parsability metric for word sequences. He reports on a simple yet practical way of identifying grammar errors. The method is particularly useful for discovering systematic problems in a large grammar with reasonable coverage. The idea behind it is that each (under-generating) error in the gram-

Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties, pages 36–44,
Sydney, July 2006.

mar leads to the parsing failure of some specific grammatical sentences. By running the grammar over a large corpus, the corpus can be split into two subsets: the set of sentences covered by the grammar and the set of sentences that failed to parse. The errors can be identified by comparing the *statistical difference* between these two sets of sentences. By *statistical difference*, any kind of uneven distribution of linguistic phenomena is meant. In the case of van Noord (2004), the word sequences are used, mainly because the cost to compute and count the word sequences is minimum. The parsability of a sequence $w_i \dots w_j$ is defined as:

$$R(w_i \dots w_j) = \frac{C(w_i \dots w_j, OK)}{C(w_i \dots w_j)} \qquad (1)$$

where $C(w_i \dots w_j)$ is the number of sentences in which the sequence $w_i \dots w_j$ occurs, and $C(w_i \dots w_j, OK)$ is the number of sentences with a successful parse which contain the sequence. A frequency cut is used to eliminate the infrequent sequences. With suffix arrays and perfect hashing automata, the parsability of all word sequences (with arbitrary length) can be computed efficiently. The word sequences are then sorted according to their parsabilities. Those sequences with the lowest parsabilities are taken as direct indication of grammar errors.

Among them, one common error, and subsequently very common cause of parse failure is due to Multiword Expressions (MWEs), like phrasal verbs (*break down*), collocations (*bread and butter*), compound nouns (*coffee machine*), determiner-less PPs (*in hospital*), as well as so-called "frozen expressions" (*by and large*), as discussed by both Baldwin et al. (2004) and van Noord (2004). Indicatively, in the experiments reported in Baldwin et al. (2004), for instance, from all the errors due to missing lexical entries, one fifth were due to missing MWEs (8% of total errors). If an MWE is syntactically marked, the standard grammatical rules and lexical entries cannot generate the string, as for instance in the case of a phrasal verb like *take off*, even if the individual words that make up the MWE are contained in the lexicon.

In this paper we investigate semi-automatic methods for error mining and detection of missing lexical entries, following van Noord (2004), with the subsequent handling of the MWEs among them. The output of the error mining phase proposes a set of n-grams, which also contain MWEs. Therefore, the task is to distinguish the MWEs from the other cases. To do this, first we propose to use the World Wide Web as a very large corpus from which we collect evidence that enables us to rule out noisy cases (due to spelling errors, for instance), following Grefenstette (1999), Keller et al. (2002), Kilgarriff and Grefenstette (2003) and Villavicencio (2005). The candidates that are kept can be semi-automatically included in the grammar, by employing a lexical type predictor, whose output we use in order to add lexical entries to the lexicon, with a possible manual check by a grammar writer. This procedure significantly speeds up the process of grammar development, relieving the grammar developer of some of the burden by automatically detecting parse failures and providing semi-automatic means for handling them.

The paper starts with a discussion of MWEs and of some of the characteristics that make them so challenging for NLP, in section 2. This is followed by a more detailed discussion of the technique employed for error detection, in section 3. The approach used for distinguishing noisy sequences from MWE-related constructions using the World Wide Web is then presented. How this information is used for extending the grammar and the results obtained are then addressed in section 5.

2 Multiword Expressions

The term Multiword Expressions (MWEs) has been used to describe expressions for which the syntactic or semantic properties of the whole expression cannot be derived from its parts ((Sag et al., 2002), (Villavicencio et al., 2005)), including a large number of related but distinct phenomena, such as phrasal verbs (e.g. *come along*), nominal compounds (e.g. *frying pan*), institutionalised phrases (e.g. *bread and butter*), and many others. They are used frequently in language, and in English, Jackendoff (1997) estimates the number of MWEs in a speaker's lexicon to be comparable to the number of single words. This is reflected in several existing grammars and lexical resources, where almost half of the entries are Multiword Expressions. However, due to their heterogeneous characteristics, MWEs present a tough challenge for both linguistic and computational work (Sag et al., 2002). Some MWEs are fixed, and do not present internal variation, such as *ad*

hoc, while others allow different degrees of internal variability and modification, such as *touch a nerve* (*touch/find a nerve*) and *spill beans* (*spill several/musical/mountains of beans*). In terms of semantics, some MWEs are more opaque in their meaning (e.g. *to kick the bucket* as *to die*), while others have more transparent meanings that can be inferred from the words in the MWE (e.g. *eat up*, where the particle *up* adds a completive sense to *eat*). Therefore, to provide a unified account for the detection of these distinct but related phenomena is a real challenge for NLP systems.

3 Detection of Errors: Overview

van Noord (2004) reports on various errors that have been discovered for the Dutch Alpino Grammar (Bouma et al., 2001) semi-automatically, using the Twente Nieuws Corpus. The idea pursued by van Noord (2004) has been to locate those n-grams in the input that might be the cause of parsing failure. By processing a huge amount of data, the parsability metrics briefly presented in section 1 have been used to successfully locate various errors introduced by the tokenizer, erroneous/incomplete lexical descriptions, frozen expressions with idiosyncratic syntax, or incomplete grammatical descriptions. However, the recovery of these errors has been shown to still require significant efforts from the grammar developer. Moreover, there is no concrete data given about the distribution of the different types of errors discovered.

As also mentioned before, among the n-grams that usually cause parse failures, there is a large number of missing MWEs in the lexicon such as phrasal verbs, collocations, compound nouns, frozen expressions (e.g. *by and large*, *centre of attention*, *put forward by*, etc).

For the purpose of the detection of MWEs, we are interested in seeing what the major types of error for a typical large-scale deep grammar are. In this context, we have run the error mining experiment reported by van Noord with the English Resource Grammar (ERG; (Flickinger, 2000))[1] and the British National Corpus 2.0 (BNC; (Burnard, 2000)).

We have used a subset of the BNC written component. The sentences in this collection contain no more than 20 words and only ASCII characters. That is about 1.8M distinct sentences.

These sentences have then be fed into an efficient HPSG parser (PET; (Callmeier, 2000)) with ERG loaded. The parser has been configured with a maximum edge number limit of 100K and has run in the *best-only* mode so that it does not exhaustively find all the possible parses. The result of each sentence is marked as one of the following four cases:

- P means at least one parse is found for the sentence;
- L means the parser halted after the morphological analysis and has not been able to construct any lexical item for the input token;
- N means the search has finished normally and there is no parse found for the sentence;
- E means the search has finished abnormally by exceeding the edge number limit.

It is interesting to notice that when the ambiguity packing mechanism (Oepen and Carroll, 2000) is used and the unpacking is turned off [2], E does not occur at all for our test corpus. Running the parsability checking over the entire collection of sentences has taken the parser less than 2 days on a 64bit machine with 3GHz CPU. The results are shown in Table 1.

Result	# Sentences	Percentage
P	644,940	35.80%
L	969,452	53.82%
N	186,883	10.38%

Table 1: Distribution of Parsing Results

¿From the results shown in Table 1, one can see that ERG has full lexical span for less than half of the sentences. For these sentences, about 80% are successfully parsed. These numbers show that the grammar coverage has a significant improvement as compared to results reported by Baldwin et al. (2004) and Zhang and Kordoni (2006), mainly attributed to the increase in the size of the lexicon and the new rules to handle punctuations and fragments.

Obviously, L indicates the unknown words in the input sentence. But for N, it is not clear where

[1] ERG is a large-scale HPSG grammar for English. In this paper, we have used the January 2006 release of the grammar.

[2] For the experiment of error mining, only the parsability checking is necessary. There is no need to record the exact parses.

and what kind of error has occurred. In order to pinpoint the errors, we used the error mining techniques proposed by van Noord (2004) on the grammar and corpus. We have taken the sentences marked as N (because the errors in L sentences are already determined) and calculate the word sequence parsabilities against the sentences marked as P. The frequency cut is set to be 5. The whole process has taken no more than 20 minutes, resulting in total the parsability scores for 35K n-grams (word sequences). The distribution of n-grams in length with parsability below 0.1 is shown in Table 2.

	Number	Percentage
uni-gram	798	20.84%
bi-gram	2,011	52.52%
tri-gram	937	24.47%

Table 2: Distribution of N-gram in Length in Error Mining Results ($R(x) < 0.1$)

Although pinpointing the problematic n-grams still does not tell us what the exact errors are, it does shed some light on the cause. From Table 2 we see quite a lot of uni-grams with low parsabilities. Table 3 gives some examples of the word sequences. By intuition, we make the bold assumption that the low parsability of uni-grams is caused by the missing appropriate lexical entries for the corresponding word.[3]

For the bi-grams and tri-grams, we do see a lot of cases where the error can be repaired by just adding a multiword lexical entry into the grammar.

N-gram	Count
professionals	248
the flat	62
indication of	21
tone of voice	19
as always is	7

Table 3: Some Examples of the N-grams in Error Mining Results

In order to distinguish those n-grams that can be added into the grammar as MWE lexical entries from the other cases, we propose to validate them using evidence collected from the World Wide Web.

[3]It has later been confirmed with the grammar developer that almost all of the errors detected by these low parsability uni-grams can be fixed by adding correct lexical entries.

4 Detection of MWEs and related constructions

Recently, many researchers have started using the World Wide Web as an extremely large corpus, since, as pointed out by Grefenstette (1999), the Web is the largest data set available for NLP ((Grefenstette, 1999), (Keller et al., 2002), (Kilgarriff and Grefenstette, 2003) and (Villavicencio, 2005)). For instance, Grefenstette employs the Web to do example-based machine translation of compounds from French into English. The method he employs would suffer considerably from data sparseness, if it were to rely only on corpus data. So for compounds that are sparse in the BNC he also obtains frequencies from the Web. The scale of the Web can help to minimise the problem of data sparseness, that is especially acute for MWEs, and Villavicencio (2005) uses the Web to find evidence to verify automatically generated VPCs. This work is built on these, in that we propose to employ the Web as a corpus, using frequencies collected from the Web to detect MWEs among the n-grams that cause parse failure. We concentrate on the 482 most frequent candidates, to verify t he method.

The candidate list has been pre-processed to remove systematic unrelated entries, like those including acronyms, names, dates and numbers, following Bouma and Villada (2002). Using Google as a search engine, we have looked for evidence on the Web for each of the candidate MWEs, that have occurred as an exact match in a webpage. For each candidate searched, Google has provided us with a measure of frequency in the form of the number of pages in which it appears. Table 4 shows the 10 most frequent candidates, and among these there are parts of formulae, frozen expressions and collocations. Table 5 on the other hand, shows the 10 least frequent candidates. From the total of candidates, 311 have been kept while the other have been discarded as noise.

A manual inspection of the candidates has revealed that indeed the list contains a large amount of MWEs and frozen expressions like *taking into account the, good and evil, by and large, put forward by* and *breach of contract*. Some of these cases, like *come into effect in*, have very specific subcategorisation requirements, and this is reflected by the presence of the prepositions *into and in* in the ngram. Other cases seem to be part of formulae, like *but also in*, as part of *not only X but*

Table 4: Top 10 Candidate Multiword Expressions

MWE	Pages	Entropy	Prob(%)
the burden of	36600000	0.366	79.4
and cost effective	34400000	0.372	70.7
the likes of	34400000	0.163	93.1
but also in	27100000	0.038	98.9
to bring together	25700000	0.086	96.6
points of view	24500000	0.017	99.6
and the more	23700000	0.512	61.5
with and without	23100000	0.074	97.4
can do for	22300000	0.003	99.9
taking into account the	22100000	0.009	99.6
but what about	21000000	0.045	98.7
the ultimate in	17400000	0.199	90.0

Table 5: Bottom 10 Candidate Multiword Expressions

MWE	Pages	Entropy	Prob (%)
stand by and	1350000	0.399	65.5
discharged from hospital	553000	0.001	99.9
shock of it	92300	0.541	44.6
was woken by	91400	0.001	99.9
telephone rang and	43700	0.026	99.2
glanced across at	36900	0.003	99.9
the citizens charter	22900	0.070	97.9
input is complete	13900	0.086	97.2
from of government	706	0.345	0.1
the to infinitive	561	0.445	1.4

also Y, but what about, and *the more the* (part of *the more the Yer*).

However, among the candidates there still remain those that are not genuine MWEs, like *of alcohol and* and *than that in*, which contain very frequent words that enable them to obtain a very high frequency count without being an MWE. Therefore, to detect these cases, the remainder of the candidates could be further analysed using some statistical techniques to try to distinguish them from the more likely MWEs among the candidates. This is done by Bouma and Villada (2002) who investigated some measures that have been used to identify certain kinds of MWEs, focusing on collocational prepositional phrases, and on the tests of mutual information, log likelihood and χ^2. One significant difference here is that this work is not constrained to a particular type of MWEs, but has to deal with them in general. Moreover, the statistical measures used by Bouma and Villada demand the knowledge of single word frequencies which can be a problem when using Google especially for common words like *of* and *a*.

In Tables 4 and 5 we present two alternative measures that combined can help to detect false candidates. The rational is similar to the statistical tests, without the need of searching for the frequency of each of the words that make up the MWE. We assume that if a candidate is just a result of the random occurrence of very frequent words most probably the order of the words in the ngram is not important. Therefore, given a candidate, such as *the likes of*, we measure the frequency of occurrence of all its permutations (e.g. *the of likes, likes the of, etc*) and we calculate the candidate's entropy as

$$S = -\frac{1}{\log N} \sum_{k=1}^{N} P_i \ \log P_i \qquad (2)$$

where P_i is the probability of occurrence of a given permutation, and N the total number of permutations. The entropy above defined has its maximum at $S = 1$ when all permutations are equally probably, which indicates a clear signature of a random nature. On the other hand, when order is very important and only a single configuration is allowed the entropy has its minimum, $S = 0$. An ngram with low entropy has good chances of being an MWE. A close inspection on Table 4 shows that the top two candidate ngrams have relatively high entropies (here we consider high entropy when $S > 0.3$). In the first case this can be explained by the fact that the word *the* can appear after the word *of* without compromising the MWE meaning as in *the burden of the job*. In the second case it shows that the real MWE is *cost effective* and the word *and* can be either in the beginning or in the end of the trigram. In fact for a trigram with only two acceptable permutations the entropy is $S = \log 2 / \log 6 \simeq 0.39$, very close to what is obtained .

We also show the probability of occurrence of each candidate ngram among its permutations (P_1). Most of the candidates in the list are more frequent than their permutations. In Table 4 we find two exceptions which are clearly spelling errors in the last 2 ngrams. Therefore low P_1 can be a good indicative of a noisy candidate. Another good predictor is the relative frequency between the candidates. Given the occurrence values for the most frequent candidates, we consider that by using a threshold of 20,000 occurrences, it is possible to remove the more noisy cases.

We note that the grammar can also impose some restrictions in the order of the elements in the ngram, in the sense that some of the generated permutations are ungrammatical (e.g. *the of likes*) and will most probably have null or very low frequencies. Therefore, on top of the constraints on the lexical order there are also constraints on the constituent order of a candidate which will be reflected in these measures.[4]

The remainder candidates can be semi-automatically included in the grammar, by using a lexical type predictor, as described in the next section. With this information, each candidate is added as a lexical entry, with a possible manual check by a grammar writer prior to inclusion in the grammar.

[4]Google ignores punctuation between the elements of the ngram. This can lead to some hits being returned for some of the ungrammatical permuted ngrams, such as *one one by* in the sentence *We're going to catch people one by one. One day,...* from www.beertravelers.com/lists/drafttech.html. On the other hand, Google only returns the number of pages where a given ngram occurred, but not the number of times it occurred in that page. This can result in a huge underestimation especially for very frequent ngrams and words, which can be used mo re than once in a given page. Therefore, a conservative view of these frequencies must be adopted, given that for some ngrams they might be inflated and for others deflated.

5 Automated Deep Lexical Acquisition

In section (3), we have seen that more than 50% of the sentences contain one or more unknown words. And about half of the other parsing failures are also due to lexicon missing. In this section, we propose a statistical approach towards lexical type prediction for unknown words, including multi-word expressions.

5.1 Atomic Lexical Types

Lexicalist grammars are normally composed of a limited number of rules and a lexicon with rich linguistic features attached to each entry. Some grammar formalisms have a type inheriting system to encode various constraints, and a flat structure of the lexicon with each entry mapped onto one type in the inheritance hierarchy. The following discussion is based on *Head-driven Phrase Structure Grammar (HPSG)* (Pollard and Sag, 1994), but should be easily adapted to other formalisms, as well.

The lexicon of HPSG consists of a list of well-formed *Typed Feature Structures (TFSs)* (Carpenter, 1992), which convey the constraints on specific words by two ways: the type compatibility, and the feature-value consistency. Although it is possible to use both features and types to convey the constraints on lexical entries, large grammars prefer the use of types in the lexicon because the inheritance system prevents the redundant definition of feature-values. And the feature-value constraints in the lexicon can be avoided by extending the types. Say we have n lexical entries $L_i : \underset{t}{[\mathrm{F} \quad a_1]} \ldots \; L_n : \underset{t}{[\mathrm{F} \quad a_n]}$. They share the same lexical type t, but take different values for the feature F. If $a_1, \ldots, a_n$ are the only possible values for F in the context of type t, we can extend the type t with subtypes $t_{a1} : \underset{t}{[\mathrm{F} \quad a_1]} \ldots \; t_{an} : \underset{t}{[\mathrm{F} \quad a_n]}$ and modify the lexical entries to use these new types, respectively. Based on the fact that large grammars normally have a very restricted number of feature-values constraints for each lexical type, the increase of the types is acceptable. It is also typical that the types assigned to lexical entries are maximum on the type hierarchy, which means that they have no further subtypes. We will call the maximum lexical types after extension the *atomic lexical types*. Then the lexicon will be a multi-valued mapping from the word stems to the atomic lexical types.

Needless to underline here that all we have mentioned above is not applicable exclusively to HPSG, but to many other formalisms based on TFSs, which makes our assumptions about atomic lexical types all the more relevant for a wide range of systems and applications.

5.2 Statistical Lexical Type Predictor

Given that the lexicon of deep grammars can be modelled by a mapping from word stems to atomic lexical types, we now go on designing the statistical methods that can automatically "guess" such mappings for unknown words.

Similar to Baldwin (2005), we also treat the problem as a classification task. But there is an important difference. While Baldwin (2005) makes predictions for each unknown word, we create a new lexical entry for each occurrence of the unknown word. The assumption behind this is that there should be exactly one lexical entry that corresponds to the occurrence of the word in the given context[5].

We use a single classifier to predict the atomic lexical type. There are normally hundreds of atomic lexical types for a large grammar. So the classification model should be able to handle a large number of output classes. We choose the Maximum Entropy-based model because it can easily handle thousands of features and a large number of possible outputs. It also has the advantages of general feature representation and no independence assumption between features. With the efficient parameter estimation algorithms discussed by Malouf (2002), the training of the model is now very fast.

For our prediction model, the probability of a lexical type t given an unknown word and its context c is:

$$p(t|c) = \frac{exp(\sum_i \theta_i f_i(t,c))}{\sum_{t' \in T} exp(\sum_i \theta_i f_i(t',c))} \qquad (3)$$

where feature $f_i(t,c)$ may encode arbitrary characteristics of the context. The parameters $< \theta_1, \theta_2, \ldots >$ can be evaluated by maximising the pseudo-likelihood on a training corpus (Malouf, 2002). The detailed design and feature selection for the lexical type predictor are described in Zhang and Kordoni (2006).

[5]Lexical ambiguity is not considered here for the unknowns. In principle, this constraint can be relaxed by allowing the classifier to return more than one results by, setting a confidence threshold, for example.

In the experiment described here, we have used the latest version of the Redwoods Treebank in order to train the lexical type predictor with morphological features and context words/POS tags features [6]. We have then extracted from the BNC 6248 sentences, which contain at least one of the 311 MWE candidates verified with World Wide Web in the way described in the previous section. For each occurrence of the MWE candidates in this set of sentences, our lexical type predictor has predicted a lexical entry candidate. This has resulted in 1936 distinct entries. Only those entries with at least 5 counts have been added into the grammar. This has resulted in an extra 373 MWE lexical entries for the grammar.

This addition to the grammar has resulted in a significant increase in coverage (table 6) of 14.4%. This result is very promising, as only a subset of the candidate MWEs has been analysed, and could result in an even greater increase in coverage, if these techniques were applied to the complete set of candidates.

However, we should also point out that the coverage numbers reported in Table 6 are for a set of "difficult" sentences which contains a lot of MWEs. When compared to the numbers reported in Table 1, the coverage of the parser on this data set after adding the MWE entries is still significantly lower. This indicates that not all the MWEs can be correctly handled by simply adding more lexical entries. Further investigation is still required.

6 Conclusions

One of the important challenges for robust natural language processing systems is to be able to deal with the systematic parse failures caused in great part by Multiword Expressions and related constructions. Therefore, in this paper we have proposed an approach for the semi-automatic extension of grammars by using an error mining technique for the detection of MWE candidates in texts and for predicting possible lexico-syntactic types for them. The approach presented is based on that of van Noord (2004) and proposes a set of MWE candidates. For this set of candidates, using the World Wide Web as a large corpus, frequencies are gathered for each candidate. These in conjunction with some statistical measures are employed for ruling out noisy cases like spelling mistakes (*from of government*) and frequent non-MWE sequences like *input is complete*.

With this information the remaining sequences are analysed by a statistical type predictor that assigns the most likely lexical type for each of the candidates in a given context. By adding these to the grammar as new lexical entries, a considerable increase in coverage of 14.4% was obtained.

The approach proposed employs simple and self-contained techniques that are language-independent and can help to semi-automatically extend the coverage of a grammar without relying on external resources, like electronic dictionaries and ontologies that are expensive to obtain and not available for all languages. Therefore, it provides an inexpensive and reusable manner of helping and speeding up the grammar engineering process, by relieving the grammar developer of some of the burden of extending the coverage of the grammar.

As future work we intend to investigate further statistical measures that can be applied robustly to different types of MWEs for refining even more the list of candidates and distinguishing false positives, like *of alcohol and* from MWEs, like *put forward by*. The high frequency with which the former occur in corpora and the more accute problem of data sparseness that affects the latter make this a difficult task.

[6] The POS tags are produced with the *TnT* tagger.

References

Timothy Baldwin, Emily M. Bender, Dan Flickinger, Ara Kim, and Stephan Oepen. 2004. Road-testing the English Resource Grammar over the British National Corpus. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.

Timothy Baldwin. 2005. Bootstrapping deep lexical resources: Resources for courses. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*, pages 67–76, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Gosse Bouma and Begońa Villada. 2002. Corpus-based acquisition of collocational prepositional phrases. In *Proceedings of the Computational Linguistics in the Netherlands (CLIN) 2001*, University of Twente.

Gosse Bouma, Gertjan van Noord, and Robert Malouf. 2001. Alpino: Wide-coverage computational analysis of dutch. In *Computational Linguistics in The Netherlands 2000*.

	Entries Added	Item #	Covered #	Coverage
ERG	0	6246	268	4.3%
ERG+MWE(Web)	373	6246	1168	18.7%

Table 6: Parser coverage on "difficult" sentences before/after adding MWE lexical entries

Lou Burnard. 2000. User Reference Guide for the British National Corpus. Technical report, Oxford University Computing Services.

M. Butt, S. Dipper, A. Frank, and T.H. King. 1999. Writing large-scale parallel grammars for english, french, and german. In *Proceedings of the LFG99 Conference*. CSLI Publications.

Ulrich Callmeier. 2000. PET – a platform for experimentation with efficient HPSG processing techniques. *Journal of Natural Language Engineering*, 6(1):99–108.

Bob Carpenter. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press, Cambridge, England.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

Gregory Grefenstette. 1999. The World Wide Web as a resource for example-based machine translation tasks. In *Proceedings of ASLIB, Conference on Translating and the Computer*, London.

Ray Jackendoff. 1997. Twistin' the night away. *Language*, 73:534–59.

Frank Keller, Maria Lapata, and Olga Ourioupina. 2002. Using the Web to overcome data sparseness. In Jan Hajič and Yuji Matsumoto, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 230–237, Philadelphia.

Adam Kilgarriff and Gregory Grefenstette. 2003. Introduction to the special issue on web as corpus. *Computational Linguistics*, 29.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conferencde on Natural Language Learning (CoNLL-2002)*, pages 49–55.

Stephan Oepen and John Carroll. 2000. Ambiguity packing in constraint-based parsing — practical results. In *Proceedings of the 1st Conference of the North American Chapter of the ACL*, pages 162–169, Seattle, WA.

Carl J. Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, Illinois.

Ivan Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*, pages 1–15, Mexico City, Mexico.

Gertjan van Noord. 2004. Error mining for wide-coverage grammar engineering. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 446–453, Barcelona, Spain, July.

Aline Villavicencio, Francis Bond, Anna Korhonen, and Diana McCarthy. 2005. Introduction to the special issue on multiword expressions: having a crack at a hard nut. *Journal of Computer Speech and Language Processing*, 19.

Aline Villavicencio. 2005. The availability of verb-particle constructions in lexical resources: How much is enough? *Journal of Computer Speech and Language Processing*, 19.

Yi Zhang and Valia Kordoni. 2006. Automated deep lexical acquisition for robust open texts processing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy.