Acquisition of Unknown Word Paradigms for Large-Scale Grammars

Kostadin Cholakov
University of Groningen
The Netherlands
k.cholakov@rug.nl

Gertjan van Noord
University of Groningen
The Netherlands
g.j.m.van.noord@rug.nl

Abstract

Unknown words are a major issue for large-scale grammars of natural language. We propose a machine learning based algorithm for acquiring lexical entries for all forms in the paradigm of a given unknown word. The main advantages of our method are the usage of word paradigms to obtain valuable morphological knowledge, the consideration of different contexts which the unknown word and all members of its paradigm occur in and the employment of a full-blown syntactic parser and the grammar we want to improve to analyse these contexts and provide elaborate syntactic constraints. We test our algorithm on a large-scale grammar of Dutch and show that its application leads to an improved parsing accuracy.

1 Introduction

In this paper, we present an efficient machine learning based method for automated lexical acquisition (LA) which improves the performance of large-scale computational grammars on real-life tasks.

Our approach has three main advantages which distinguish it from other methods applied to the same task. First, it enables the acquisition of the *whole paradigm* of a given unknown word while other approaches are only concerned with the particular word form encountered in the data subject to LA. Second, we analyse *different contexts* which the unknown word occurs in. Third, the analysis of these contexts is provided by a *full-blown syntactic parser* and the *grammar* we aim to improve which gives the grammar the opportunity to participate *directly* in the LA process.

Our method achieves an F-measure of 84.6% on unknown words in experiments with the wide-coverage Alpino grammar (van Noord, 2006) of Dutch. The integration of this method in the parser leads to a 4.2% error reduction in terms of labelled dependencies.

To predict a lexical entry for a given unknown word, we take into account two factors– its morphology and the syntactic constraints imposed by its context. As for the former, the acquisition of the whole paradigm provides us with a valuable source of morphological information. If we were to deal with only one form of the unknown word, this information would not be accessible.

Further, looking at different contexts of the unknown word gives us the possibility to work with linguistically diverse data and to incorporate more syntactic information into the LA process. Cases where this is particularly important include morphologically ambiguous words and verbs which subcategorize for various types of syntactic arguments. We also consider contexts of the other members of the paradigm of the unknown word in order to increase the amount of linguistic data our method has access to.

Finally, the usage of a *full-blown syntactic parser* and the *grammar* we want to acquire lexical entries for has two advantages. First, LA can benefit from the high-quality analyses such a parser produces and the elaborate syntactic information they provide. Second, this information comes directly from the grammar, thus allowing the LA process to make predictions based on what the grammar considers to be best suited for it.

Coling 2008: Poster Volume, pages 153–161,
Beijing, August 2010

The remainder of the paper is organised as follows. Section 2 describes the basic steps in our LA algorithm. Section 3 presents initial experiments conducted with Alpino and shows that the main problems our LA method encounters are the acquisition of morphologically ambiguous words, the learning of the proper subcategorization frames for verbs and the acquisition of particular types of adjectives. In Section 4 we make extensive use of the paradigms of the unknown words to develop specific solutions for these problems. Section 5 describes experiments with our LA method applied to a set of real unknown words. Section 6 provides a comparison between our approach and work previously done on LA. This section also discusses the application of our method to other systems and languages.

2 Basic Algorithm

The Alpino wide-coverage dependency parser is based on a large stochastic attribute value grammar. The grammar takes a 'constructional' approach, with rich lexical representations stored in the lexicon and a large number of detailed, construction specific rules (about 800). Currently, the lexicon contains about 100K lexical entries and a list of about 200K named entities. Each word is assigned one or more lexical types. For example, the verb *amuseert* (to amuse) is assigned two lexical types– *verb(hebben,sg3,intransitive)* and *verb(hebben,sg3,transitive)*– because it can be used either transitively or intransitively. The other type features indicate that it is a present third person singular verb and it forms perfect tense with the auxiliary verb *hebben*.

The goal of our LA method is to *assign* the correct lexical type(s) to a given unknown word. The method takes into account only open-class lexical types: nouns, adjectives and verbs, under the assumption that the grammar is already able to handle all closed-class cases. We call the types considered by our method *universal types*. The adjectives can be used as adverbs in Dutch and thus, we do not consider the latter to be an open class.

We employ a ME-based classifier which, for some unknown word, takes various morphological and syntactic features as input and outputs lexical types. The probability of a lexical type *t*, given an unknown word and its context *c* is:

$$(1) \qquad p(t|c) = \frac{exp(\sum_i \Theta_i f_i(t,c))}{\sum_{t' \in T} exp(\sum_i \Theta_i f_i(t',c))}$$

where $f_i(t, c)$ may encode arbitrary characteristics of the context and $< \Theta_1, \Theta_2, ... >$ can be evaluated by maximising the pseudo-likelihood on a training corpus (Malouf, 2002).

Table 1 shows the features for the noun *inspraakprocedures* (consultation procedures). Row **(i)** contains 4 separate features derived from the prefix of the word and 4 other suffix features are given in row **(ii)**. The two features in rows **(iii)** and **(iv)** indicate whether the word starts with a particle and if it contains a hyphen, respectively.

Another source of morphological features is the paradigm of the unknown word which provides information that is otherwise inaccessible. For example, in Dutch, neuter nouns always take the *het* definite article while all other noun forms are used with the *de* article. Since the article is distinguishable only in the singular noun form, the correct article of a word, assigned a plural noun type, can be determined if we know its singular form.

We adopt the method presented in Cholakov and van Noord (2009) where a finite state morphology is applied to generate the paradigm(s) of a given word. The morphology does not have access to any additional linguistic information and thus, it generates all possible paradigms allowed by the word structure. Then, the number of search hits Yahoo returns for each form in a given paradigm is combined with some simple heuristics to determine the correct paradigm(s).

However, we make some modifications to this method because it deals only with *regular* morphological phenomena. Though all typical irregularities are included in the Alpino lexicon, there are cases of irregular verbs composed with particles which are not listed there. One such example is the irregular verb *meevliegen* (to fly with someone) for which no paradigm would be generated.

To avoid this, we use a list of common particles to strip off any particle from a given unknown word. Once we have removed a particle, we check if what is left from the word is listed in the lexicon as a verb (e.g. *vliegen* in the case of ***mee****vliegen*). If so, we extract all members of its paradigm from

Features
i) i, in, ins, insp
ii) s, es, res, ures
iii) particle_yes #in this case *in*
iv) hyphen_no
v) noun⟨de,pl⟩
vi) noun(de,count,pl), tmp_noun(de,count,sg)
vii) noun(de), noun(count), noun(pl), tmp_noun(de) tmp_noun(count), tmp_noun(sg)

Table 1: Features for *inspraakprocedures*

the lexicon and use them to build the paradigm of the unknown word. All forms are validated by using the same web-based heuristics as in the original model of Cholakov and van Noord (2009).

A single paradigm is generated for *inspraakprocedures* indicating that this word is a plural *de* noun. This information is explicitly used as a feature in the classifier which is shown in row **(v)** of Table 1.

Next, we obtain syntactic features for *inspraakprocedures* by extracting a number of sentences which it occurs in from large corpora or Internet. These sentences are parsed with a different 'mode' of Alpino where this word is assigned all universal types, i.e. it is treated as being maximally *ambiguous*. For each sentence only the best parse is preserved. Then, the lexical type that has been assigned to *inspraakprocedures* in this parse is stored. During parsing, Alpino's POS tagger (Prins and van Noord, 2001) keeps filtering implausible type combinations. For example, if a determiner occurs before the unknown word, all verb types are typically not taken into consideration. This heavily reduces the computational overload and makes parsing with universal types computationally feasible. When all sentences have been parsed, a list can be drawn up with the types that have been used and their frequency:

(2)
noun(de,count,pl) 78
tmp_noun(de,count,sg) 7
tmp_noun(het,count,pl) 6
proper_name(pl,'PER') 5
proper_name(pl,'ORG') 3
verb(hebben,pl,vp) 1

The lexical types assigned to *inspraakprocedures* in at least 80% of the parses are used as features in the classifier. These are the two features in row **(vi)** of Table 1. Further, as illustrated in row **(vii)**, each attribute of the considered types is also taken as a separate feature. By doing this, we let the grammar decide which lexical type is best suited for a given unknown word. This is a new and effective way to include the *syntactic constraints* of the context in the LA process.

However, for the parsing method to work properly, the disambiguation model of the parser needs to be adapted. The model heavily relies on the lexicon and it has learnt preferences how to parse certain phrases. For example, it has learnt a preference to parse prepositional phrases as verb complements, if the verb includes such a subcategorization frame. This is problematic when parsing with universal types. If the unknown word is a verb and it occurs together with a PP, it would always get analysed as a verb which subcategorizes for a PP.

To avoid this, the disambiguation model is retrained on a specific set of sentences meant to make it more robust to input containing many unknown words. We have selected words with low frequency in large corpora and removed them temporarily from the Alpino lexicon. Less frequent words are typically not listed in the lexicon and the selected words are meant to simulate their behaviour. Then, all sentences from the Alpino treebank which contain these words are extracted and used to retrain the disambiguation model.

3 Initial Experiments and Evaluation

To evaluate the performance of the classifier, we conduct an experiment with a target type inventory of 611 universal types. A type is considered universal only if it is assigned to at least 15 distinct words occurring in large Dutch newspaper corpora ($\sim$16M sentences) automatically parsed with Alpino.

In order to train the classifier, 2000 words are temporarily removed from the Alpino lexicon. The same is done for another 500 words which are used as a test set. All words have between 50 and 100 occurrences in the corpora. This selection is again meant to simulate the behaviour of unknown words. Experiments with a minimum lower than 50 occurrences have shown that this is a reasonable threshold to filter out typos, words written together, etc.

The classifier yields a probability score for each predicted type. Since a given unknown word can have more than one correct type, we want to predict multiple types. However, the least frequent types, accounting together for less than 5% of probability mass, are discarded.

We evaluate the results in terms of precision and recall. Precision indicates how many types found by the method are correct and recall indicates how many of the lexical types of a given word are actually found. The presented results are the average precision and recall for the 500 test words.

Additionally, there are three baseline methods:

- *Naive*– each unknown word is assigned the most frequent type in the lexicon: *noun(de,count,sg)*
- *POS tagger*– the unknown word is given the type most frequently assigned by the Alpino POS tagger in the parsing stage
- *Alpino*– the unknown word is assigned the most frequently used type in the parsing stage

The overall results are given in Table 2. Table 3 shows the results for each POS in our model.

Model	Precision(%)	Recall(%)	F-measure(%)
Naive	19.60	18.77	19.17
POS tagger	30	26.21	27.98
Alpino	44.60	37.59	40.80
Our model	86.59	78.62	82.41

Table 2: Overall experiment results

POS	Precision(%)	Recall(%)	F-measure(%)
Nouns	93.83	88.61	91.15
Adjectives	75.50	73.12	74.29
Verbs	77.32	55.37	64.53

Table 3: Detailed results for our model

Our LA method clearly improves upon the baselines. However, as we see in Table 3, adjectives and especially verbs remain difficult to predict.

The problems with the former are due to the fact that Alpino employs a rather complicated adjective system. The classifier has difficulties distinguishing between 3 kinds of adjectives: **i)** adjectives which can attach to and modify verbs and verbal phrases (VPs) (3-a), **ii)** adjectives which can attach to verbs and VPs but modify one of the complements of the verb, typically the subject (3-b) and **iii)** adjectives which cannot attach to verbs and VPs (3-c).

(3)
a. De hardloper loopt *mooi*.
DET runner walks nice
'The runner runs nicely = The runner has a good running technique'
b. Hij loopt *dronken* naar huis.
he walks drunk to home
'He walks home drunk = He is walking home while being drunk'
c. *Hij loopt *nederlandstalig*.
he walks Dutch speaking
'He walks Dutch speaking.'

Each of these is marked by a special attribute in the lexical type definitions– *adv*, *padv* and *nonadv*, respectively. Since all three of them are seen in 'typical' adjectival contexts where they modify nouns, it is hard for the classifier to make a distinction. The predictions appear to be arbitrary and there are many cases where the unknown word is classified both as a *nonadv* and an *adv* adjective. It is even more difficult to distinguish between *padv* and *adv* adjectives since this is a solely semantic distinction.

The main issue with verbs is the prediction of the correct subcategorization frame. The classifier tends to predict mostly transitive and intransitive verb types. As a result, it either fails to capture infrequent frames which decreases the recall or, in cases where it is very uncertain what to predict, it assigns a lot of types that differ only in the subcat frame, thus damaging the precision. For example, *onderschrijf* ('to agree with') has 2 correct subcat frames but receives 8 predictions which differ only in the subcat features.

One last issue is the prediction, in some rare cases, of types of the wrong POS for morphologically ambiguous words. In most of these cases adjectives are wrongly assigned a past participle type but also some nouns receive verb predictions. For instance, *OESO-landen* ('countries of the OESO organisation') has one correct noun type but because *landen* is also the Dutch verb for 'to land' the classifier wrongly assigns a verb type as well.

4 Improving LA

4.1 POS Correction

Since the vast majority of wrong POS predictions has to do with the assignment of incorrect verb types, we decided to explicitly use the generated verb paradigms as a filtering mechanism. For each word which is assigned a verb type, we check if there is a verb paradigm generated for it. If not, all verb types predicted for the word are discarded.

In very rare cases a word is assigned *only* verb types and therefore, it ends up with no predictions. For such words, we examine the ranked list of predicted types yielded by the classifier and the word receives the non-verb lexical type with the highest probability score. If this type happens to be an adjective one, we first check whether there is an adjective paradigm generated for the word in question. If not, the word gets the noun type with the highest probability score.

The same procedure is also applied to all words which are assigned an adjective type. However, it is not used for words predicted to be nouns because the classifier is already very good at predicting nouns. Further, the generated noun paradigms are not reliable enough to be a filtering mechanism because there are mass nouns with no plural forms and thus with no paradigms generated.

Another modification we make to the classifier output has to do with the fact that past participles (psp) in Dutch can also be used as adjectives. This systematic ambiguity, however, is not treated as such in Alpino. Each psp should also have a separate adjective lexical entry but this is not always the case. That is why, in some cases, the classifier fails to capture the adjective type of a given psp. To account for it, all words predicted to be past participles but not adjectives are assigned two additional adjective types– one with the *nonadv* and one with the *adv* feature. For reasons explained later on, a type with the *padv* feature is not added.

After the application of these techniques, all cases of words wrongly predicted to be verbs or adjectives have been eliminated.

4.2 Guessing Subcategorization Frames

Our next step is to guess the correct subcategorization feature for verbs. Learning the proper subcat frame is well studied (Brent, 1993; Manning, 1993; Briscoe and Caroll, 1997; Kinyon and Prolo, 2002; O'Donovan et al., 2005). Most of the work follows the 'classical' Briscoe and Caroll (1997) approach where the verb and the subcategorized complements are extracted from the output analyses of a probabilistic parser and stored as syntactic patterns. Further, some statistical techniques are applied to select the most probable frames out of the proposed syntactic patterns.

Following the observations made in Korhonen et al. (2000), Lapata (1999) and Messiant (2008), we employ a maximum likelihood estimate (MLE) from observed relative frequencies with an empirical threshold to filter out low probability frames. For each word predicted to be a verb, we look up the verb types assigned to it during the parsing with universal types. Then, the MLE for each subcat frame is determined and only frames with MLE of 0.2 and above are considered. For example, *jammert* (to moan.3SG.PRES) is assigned a single type– *verb(hebben,sg3,intransitive)*. However, the correct subcat features for it are *intransitive* and *sbar*. Here is the list of all verb types assigned to *jammert* during the parsing with universal types:

(4)
verb(hebben,sg3,intransitive) 48
verb(hebben,sg3,transitive) 15
verb(hebben,past(sg),np_sbar) 3
verb(hebben,past(sg),tr_sbar) 3
verb(zijn,sg3,intransitive) 2
verb(hebben,past(sg),ld_pp) 2
verb(hebben,sg3,sbar) 1

The MLE for the intransitive subcat feature is 0.68 and for the transitive one– 0.2. All previously predicted verb types are discarded and each considered subcat frame is used to create a new lexical type. That is how *jammert* gets two types at the end– the correct *verb(hebben,sg3,intransitive)* and the incorrect *verb(hebben,sg3,transitive)*. The *sbar* frame is wrongly discarded.

To avoid such cases, the generated word paradigms are used to increase the number of contexts observed for a given verb. Up to 200 sentences are extracted for each form in the paradigm of a given word predicted to be a verb. These sentences are again parsed with the universal types and then, the MLE for each subcat frame is recal-

culated.

We evaluated the performance of our MLE-based method on the 116 test words predicted to be verbs. We extracted the subcat features from their type definitions in the Alpino lexicon to create a gold standard of subcat frames. Additionally, we developed two baseline methods: **i)** all frames assigned during parsing are considered and **ii)** each verb is taken to be both transitive and intransitive. Since most verbs have both or one of these frames, the purpose of the second baseline is to see if there is a simpler solution to the problem of finding the correct subcat frame. The results are given in Table 4.

Model	Precision(%)	Recall(%)	F-measure(%)
all frames	16.76	94.34	28.46
tr./intr.	62.29	69.17	65.55
our model	85.82	67.28	75.43

Table 4: Subcat frames guessing results

Our method significantly outperforms both baselines. It is able to correctly identify the transitive and/or the intransitive frames. Since they are the most frequent ones in the test data, this boosts up the precision. However, the method is also able to capture other, less frequent subcat frames. For example, after parsing the additional sentences for *jammert*, the *sbar* frame had enough occurrences to get above the threshold. The MLE for the transitive one, on the other hand, fell below 0.2 and it was correctly discarded.

4.3 Guessing Adjective Types

We follow a similar approach for finding the correct adjective type. It should be noted that the distinction among *nonadv*, *adv* and *padv* does not exist for every adjective form. Most adjectives in Dutch get an *-e* suffix when used attributively– *de mooi**e**/mooier**e**/mooist**e** jongen* (the nice/nicer/nicest boy). Since these inflected forms can only occur before nouns, the distinction we are dealing with is not relevant for them. Thus we are only interested in the noninflected base, comparative and superlative adjective forms.

One of the possible output formats of Alpino is dependency triples. Here is the output for the sentence in (3-a):

(5)
```
verb:loop|hd/su|noun:hardloper
noun:hardloper|hd/det|det:de
verb:loop|hd/mod|adj:mooi
verb:loop|–/–|punct:.
```

Each line is a single dependency triple. The line contains three fields separated by the '|' character. The first field contains the root of the head word and its POS, the second field indicates the type of the dependency relation and the third one contains the root of the dependent word and its POS. The third line in (5) shows that the adjective *mooi* is a modifier of the head, in this case the verb *loopt*. Such a dependency relation indicates that this adjective can modify a verb and therefore, it belongs to the *adv* type.

As already mentioned, *padv* adjectives cannot be distinguished from the ones of the *adv* kind. That is why, if the classifier has decided to assign a *padv* type to a given unknown word, we discard all other adjective types assigned to it (if any) and do not apply the technique described below to this word.

For each of the 59 words assigned an non-inflected adjective type after the POS correction stage, we extract up to 200 sentences for all non-inflected forms in its paradigm. These sentences are parsed with Alpino and the universal types and the output is dependency triples. All triples where the unknown word occurs as a dependent word in a head modifier dependency (*hd/mod*, as shown in (5)) and its POS is adjective are extracted from the parse output. We calculate the MLE of the cases where the head word is a verb, i.e. where the unknown word modifies a verb. If the MLE is 0.05 or larger, the word is assigned an *adv* lexical type.

For example, the classifier correctly identifies the word *doortimmerd* (solid) as being of the *adjective(no_e(nonadv))* type but it also predicts the *adjective(no_e(adv))*[1] type for it. Since we have not found enough sentences where this word modifies a verb, the latter type is correctly discarded. Our technique produced correct results for 53 out of the 59 adjectives processed.

[1]The *no_e* type attribute denotes a noninflected base adjective form.

4.4 Improved Results and Discussion

Table 5 presents the results obtained after applying the improvement techniques described in this section to the output of the classifier (the 'Model 2' rows). For comparison, we also give the results from Table 3 again (the 'Model 1' rows). The numbers for the nouns happen to remain unchanged and that is why they are not shown in Table 5.

POS	Models	Prec.(%)	Rec.(%)	F-meas.(%)
Adj	Model 1	75.50	73.12	74.29
	Model 2	85.16	80.16	82.58
Verbs	Model 1	77.32	55.37	64.53
	Model 2	80.56	56.24	66.24
Overall	Model 1	86.59	78.62	82.41
	Model 2	89.08	80.52	84.58

Table 5: Improved results

The automatic addition of adjective types for past participles improved significantly the recall for adjectives and our method for choosing between *adv* and *nonadv* types caused a 10% increase in precision.

However, these procedures also revealed some incomplete lexical entries in Alpino. For example, there are two past participles not listed as adjectives in the lexicon though they should be. Thus when our method *correctly* assigned them adjective types, it got punished since these types were not in the gold standard.

We see in Table 5 that the increase in precision for the verbs is small and recall remains practically unchanged. The unimproved recall shows that we have not gained much from the subcat frame heuristics. Even when the number of the observed sentences was increased, less frequent frames often remained unrecognisable from the noise in the parsed data. This could be seen as a proof that in the vast majority of cases verbs are used *transitively* and/or *intransitively*. Since the MLE method we employ proved to be good at recognising these two frames and differentiating between them, we have decided to continue using it.

The overall F-score improved by only 2% because the modified verb and adjective predictions are less than 30% of the total predictions made by the classifier.

5 Experiment with Real Unknown Words

To investigate whether the proposed LA method is also beneficial for the parser, we observe how parsing accuracy changes when the method is employed. Accuracy in Alpino is measured in terms of labelled dependencies.

We have conducted an experiment with a test set of 300 sentences which contain 188 real unknown words. The sentences have been randomly selected from the manually annotated LASSY corpus (van Noord, 2009) which contains text from various domains. The average sentence length is 26.54 tokens.

The results are given in Table 6. The standard Alpino model uses its guesser to assign types to the unknown words. Model 1 employs the trained ME-based classifier to predict lexical entries for the unknown words offline and then uses them during parsing. Model 2 uses lexical entries modified by applying the methods described in Section 4 to the output of the classifier (Model 1).

Model	Accuracy (%)	msec/sentence
Alpino	88.77	8658
Model 1	89.06	8772
Model 2	89.24	8906

Table 6: Results with real unknown words

Our LA system as a whole shows an error reduction rate of more than 4% with parse times remaining similar to those of the standard Alpino version. It should also be noted that though much of the unknown words are generally nouns, we see from the results that it makes sense to also employ the methods for improving the predictions for the other POS types. A wrong verb or even adjective prediction can cause much more damage to the analysis than a wrong noun one.

These results illustrate that the integration of our method in the parser can improve its performance on real-life data.

6 Discussion

6.1 Comparison to Previous Work

The performance of the LA method we presented in this paper can be compared to the performance

of a number of other approaches previously applied to the same task.

Baldwin (2005) uses a set of binary classifiers to learn lexical entries for a large-scale grammar of English (ERG; (Copestake and Flickinger, 2000)). The main disadvantage of the method is that it uses information obtained from secondary language resources– POS taggers, chunkers, etc. Therefore, the grammar takes no part in the LA process and the method acquires lexical entries based on incomplete linguistic information provided by the various resources. The highest F-measure (about 65%) is achieved by using features from a chunker but it is still 20% lower than the results we report here. Further, no evaluation is done on how the method affects the performance of the ERG when the grammar is used for parsing.

Zhang and Kordoni (2006) and Cholakov et al. (2008), on the other hand, include features from the grammar in a maximum entropy (ME) classifier to predict new lexical entries for the ERG and a large German grammar (GG; (Crysmann, 2003)), respectively. The development data for this method consist of linguistically annotated sentences from treebanks and the grammar features used in the classifier are derived from this annotation. However, when the method is applied to open-text unannotated data, the grammar features are replaced with POS tags. Therefore, the grammar is no longer directly involved in the LA process which affects the quality of the predictions. Evaluation on sentences containing real unknown words shows improvement of the coverage for the GG when LA is employed but the accuracy decreases by 2%. Such evaluation has not been done for the ERG. The results on the development data are not comparable with ours because evaluation is done only in terms of precision while we are also able to measure recall.

Statistical LA has previously been applied to Alpino as well (van de Cruys, 2006). However, his method employs less morphosyntactic features in comparison to our approach and does not make use of word paradigms. Further, though experiments on development data are performed on a smaller scale, the results in terms of F-measure are 10% lower than those reported in our case study. Experiments with real unknown words have not been performed.

Other, non-statistical LA methods also exist. Cussens and Pulman (2000) describe a symbolic approach which employs *inductive logic programming* and Barg and Walther (1998) and Fouvry (2003) follow a unification-based approach. However, the generated lexical entries might be both too general or too specific and it is doubtful if these methods can be used on a large scale. They have not been applied to broad-coverage grammars and no evaluation is provided.

6.2 Application to Other Systems and Languages

We stress the fact that the experiments with Alpino represent only a case study. The proposed LA method can be applied to other computational grammars and languages providing that the following conditions are fulfilled.

First, words have to be mapped onto some finite set of labels of which a subset of open-class (universal) labels has to be selected. This subset represents the labels which the ME-based classifier can predict for unknown words. Second, a (large) corpus has to be available, so that various sentences in which a given unknown word occurs can be extracted. This is crucial for obtaining different contexts in which this word is found.

Next, we need a parser to analyse the extracted sentences which allows for the syntactic constraints imposed by these contexts to be included in the prediction process.

Finally, as for the paradigm generation, the idea of combining a finite state morphology and web heuristics is general enough to be implemented for different languages. It is also important to note that the classifier allows for arbitrary combinations of features and therefore, a researcher is free to include any (language-specific) features he or she considers useful for performing LA.

We have already started investigating the applicability of our LA method to large-scale grammars of German and French and the initial experiments and results we have obtained are promising.

References

Baldwin, Tim. 2005. Bootstrapping deep lexical resources: Resources for courses. In *Proceedings of the ACL-SIGLEX 2005 Workshop on Deep Lexical Acquisition*, Ann Arbor, USA.

Barg, Petra and Markus Walther. 1998. Processing unknown words in HPSG. In *Proceedings of the 36th Conference of the ACL*, Montreal, Quebec, Canada.

Brent, Michael R. 1993. From grammar to lexicon: unsupervised learning of lexical syntax. *Computational Linguistics*, 19(2):243–262.

Briscoe, Ted and John Caroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, Washington, DC.

Cholakov, Kostadin and Gertjan van Noord. 2009. Combining finite state and corpus-based techniques for unknown word prediction. In *Proceedings of the 7th Recent Advances in Natural Language Processing (RANLP) conference*, Borovets, Bulgaria.

Cholakov, Kostadin, Valia Kordoni, and Yi Zhang. 2008. Towards domain-independent deep linguistic processing: Ensuring portability and re-usability of lexicalised grammars. In *Proceedings of COLING 2008 Workshop on Grammar Engineering Across Frameworks (GEAF08)*, Manchester, UK.

Copestake, Ann and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the 2nd International Conference on Language Resource and Evaluation (LREC 2000)*, Athens, Greece.

Crysmann, Berthold. 2003. On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP 2003*, Borovets, Bulgaria.

Cussens, James and Stephen Pulman. 2000. Incorporating linguistic constraints into inductive logic programming. In *Proceedings of the Fourth Conference on Computational Natural Language Learning*.

Fouvry, Frederik. 2003. Lexicon acquisition with a large-coverage unification-based grammar. In *Companion to the 10th Conference of EACL*, pages 87–90, Budapest, Hungary.

Kinyon, Alexandra and Carlos A Prolo. 2002. Identifying verb arguments and their syntactic function in the Penn Treebank. In *Proceedings of the 3rd International Conference on Language Resource and Evaluation (LREC 2002)*, Las Palmas de Gran Canaria, Spain.

Korhonen, Anna, Genevieve Gorell, and Diana McCarthy. 2000. Statistical filtering and subcategorization frame acquisition. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, Hong Kong, China.

Lapata, Mirella. 1999. Acquiring lexical generalizations from corpora. A case study for diathesis alternations. In *Proceedings of the 37th Annual Meeting of ACL*, Maryland, USA.

Malouf, Robert. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the 6th conference on Natural Language Learning (CoNLL-2002)*, pages 49–55, Taipei, Taiwan.

Manning, Christopher. 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In *Proceedings of the 31st Annual Meeting of ACL*, Columbus, OH.

Messiant, Cedric. 2008. A subcategorization acquisition system for French verbs. In *Proceedings of the ACL 2008 Student Research Workshop*, Columbus, OH.

O'Donovan, Ruth, Michael Burke, Aoife Cahill, Josef van Genabith, and Andy Way. 2005. Large-scale induction and evaluation of lexical resources from the Penn-II and Penn-III Treebanks. *Computational Linguistics*, 31(3):329–365.

Prins, Robbert and Gertjan van Noord. 2001. Unsupervised POS-tagging improves parcing accuracy and parsing efficiency. In *Proceedings of IWPT*, Beijing, China.

van de Cruys, Tim. 2006. Automatically extending the lexicon for parsing. In Huitnik, Janneje and Sophia Katrenko, editors, *Proceedings of the Eleventh ESSLLI Student Session*, pages 180–189.

van Noord, Gertjan. 2006. At last parsing is now operational. In *Proceedings of TALN*, Leuven, Belgium.

van Noord, Gertjan. 2009. Huge parsed corpora in LASSY. In *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories (TLT 7)*, Groningen, The Netherlands.

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