VPCTagger: Detecting Verb-Particle Constructions With Syntax-Based Methods

István Nagy T.[1] and **Veronika Vincze**[1,2]
[1]Department of Informatics, University of Szeged
Árpád tér 2., 6720 Szeged, Hungary
nistvan@inf.u-szeged.hu
[2]Hungarian Academy of Sciences, Research Group on Artificial Intelligence
Tisza Lajos krt. 103., 6720 Szeged, Hungary
vinczev@inf.u-szeged.hu

Abstract

Verb-particle combinations (VPCs) consist of a verbal and a preposition/particle component, which often have some additional meaning compared to the meaning of their parts. If a data-driven morphological parser or a syntactic parser is trained on a dataset annotated with extra information for VPCs, they will be able to identify VPCs in raw texts. In this paper, we examine how syntactic parsers perform on this task and we introduce VPCTagger, a machine learning-based tool that is able to identify English VPCs in context. Our method consists of two steps: it first selects VPC candidates on the basis of syntactic information and then selects genuine VPCs among them by exploiting new features like semantic and contextual ones. Based on our results, we see that VPCTagger outperforms state-of-the-art methods in the VPC detection task.

1 Introduction

Verb-particle constructions (VPCs) are a subclass of multiword expressions (MWEs) that contain more than one meaningful tokens but the whole unit exhibits syntactic, semantic or pragmatic idiosyncracies (Sag et al., 2002). VPCs consist of a verb and a preposition/particle (like *hand in* or *go out*) and they are very characteristic of the English language. The particle modifies the meaning of the verb: it may add aspectual information, may refer to motion or location or may totally change the meaning of the expression. Thus, the meaning of VPCs can be compositional, i.e. it can be computed on the basis of the meaning of the verb and the particle (*go out*) or it can be idiomatic; i.e. a combination of the given verb and particle results in a(n unexpected) new meaning (*do in* "kill"). Moreover, as their syntactic surface structure is very similar to verb – prepositional phrase combinations, it is not straightforward to determine whether a given verb + preposition/particle combination functions as a VPC or not and contextual information plays a very important role here. For instance, compare the following examples: *The hitman* ***did in*** *the president* and *What he* ***did in*** *the garden was unbelievable*. Both sentences contain the sequence *did in*, but it is only in the first sentence where it functions as a VPC and in the second case, it is a simple verb-prepositional phrase combination. For these reasons, VPCs are of great interest for natural language processing applications like machine translation or information extraction, where it is necessary to grab the meaning of the text.

The special relation of the verb and particle within a VPC is often distinctively marked at several annotation layers in treebanks. For instance, in the Penn Treebank, the particle is assigned a specific part of speech tag (RP) and it also has a specific syntactic label (PRT) (Marcus et al., 1993), see also Figure 1. This entails that if a data-driven morphological parser or a syntactic parser is trained on a dataset annotated with extra information for VPCs, it will be able to assign these kind of tags as well. In other words, the morphological/syntactic parser itself will be able to identify VPCs in texts.

In this paper, we seek to identify VPCs on the basis of syntactic information. We first examine how syntactic parsers perform on Wiki50 (Vincze et al., 2011), a dataset manually annotated for different types of MWEs, including VPCs. We then present our syntax-based tool called VPCTagger to identify VPCs, which consists of two steps: first, we select VPC candidates (i.e. verb-preposition/particle pairs) from the text and then we apply a machine learning-based technique to classify them as genuine VPCs or not. This

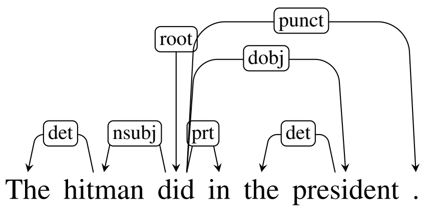

Figure 1: A dependency parse of the sentence "The hitman did in the president".

method is based on a rich feature set with new features like semantic or contextual features. We compare the performance of the parsers with that of our approach and we discuss the reasons for any possible differences.

2 Related Work

Recently, some studies have attempted to identify VPCs. For instance, Baldwin and Villavicencio (2002) detected verb-particle constructions in raw texts with the help of information based on POS-tagging and chunking, and they also made use of frequency and lexical information in their classifier. Kim and Baldwin (2006) built their system on semantic information when deciding whether verb-preposition pairs were verb-particle constructions or not. Nagy T. and Vincze (2011) implemented a rule-based system based on morphological features to detect VPCs in raw texts.

The (non-)compositionality of verb-particle combinations has also raised interest among researchers. McCarthy et al. (2003) implemented a method to determine the compositionality of VPCs and Baldwin (2005) presented a dataset in which non-compositional VPCs could be found. Villavicencio (2003) proposed some methods to extend the coverage of available VPC resources.

Tu and Roth (2012) distinguished genuine VPCs and verb-preposition combinations in context. They built a crowdsourced corpus of VPC candidates in context, where each candidate was manually classified as a VPC or not. However, during corpus building, they applied lexical restrictions and concentrated only on VPCs formed with six verbs. Their SVM-based algorithm used syntactic and lexical features to classify VPCs candidates and they concluded that their system achieved good results on idiomatic VPCs, but the classification of more compositional VPCs is more challenging.

Since in this paper we focus on syntax-based VPC identification more precisely, we also identify VPCs with syntactic parsers, it seems necessary to mention studies that experimented with parsers for identifying different types of MWEs. For instance, constituency parsing models were employed in identifying contiguous MWEs in French and Arabic (Green et al., 2013). Their method relied on a syntactic treebank, an MWE list and a morphological analyzer. Vincze et al. (2013) employed a dependency parser for identifying light verb constructions in Hungarian texts as a "side effect" of parsing sentences and report state-of-the-art results for this task.

Here, we make use of parsers trained on the Penn Treebank (which contains annotation for VPCs) and we evaluate their performance on the Wiki50 corpus, which was manually annotated for VPCs. Thus, we first examine how well these parsers identify VPCs (i.e. assigning VPC-specific syntactic labels) and then we present how VPCTagger can carry out this task. First, we select VPC candidates from raw text and then, we classify them as genuine VPCs or not.

3 Verb-particle Constructions in English

As mentioned earlier, verb-particle constructions consist of a verb and a particle. Similar constructions are present in several languages, although there might be different grammatical or orthographic norms for such verbs in those languages. For instance, in German and in Hungarian, the particle usually precedes the verb and they are spelt as one word, e.g. *aufmachen* (up.make) "to open" in German or *kinyitni* (out.open) "to open" in Hungarian. On the other hand, languages like Swedish, Norwegian, Icelandic and Italian follow the same pattern as English; namely, the verb precedes the particle and they are spelt as two words (Masini, 2005). These two typological classes require different approaches if we would like identify VPCs. For the first group, morphology-based solutions can be implemented that can identify the internal structure of compound words. For the second group, syntax-based methods can also be successful, which take into account the syntactic relation between the verb and the particle.

Many of the VPCs are formed with a motion verb and a particle denoting directions (like *go out*, *come in* etc.) and their meaning reflects this: they denote a motion or location. The meaning of VPCs belonging to this group is usually transparent and thus they can be easily learnt by second language learners. In other cases, the particle adds some aspectual information to the meaning of the verb: *eat up* means “to consume totally” or *burn out* means ”to reach a state where someone becomes exhausted”. These VPCs still have a compositional meaning, but the particle has a non-directional function here, but rather an aspectual one (cf. Jackendoff (2002)). Yet other VPCs have completely idiomatic meanings like *do up* “repair” or *do in* “kill”. In the latter cases, the meaning of the construction cannot be computed from the meaning of the parts, hence they are problematic for both language learners and NLP applications.

Tu and Roth (2012) distinguish between two sets of VPCs in their database: the *more compositional* and the *more idiomatic* ones. Differentiating between compositional and idiomatic VPCs has an apt linguistic background as well (see above) and it may be exploited in some NLP applications like machine translation (parts of compositional VPCs may be directly translated while idiomatic VPCs should be treated as one unit). However, when grouping their data, Tu and Roth just consider frequency data and treat one VPC as one lexical entry. This approach is somewhat problematic as many VPCs in their dataset are highly ambiguous and thus may have more meanings (like *get at*, which can mean “criticise”, “mean”, “get access”, “threaten”) and some of them may be compositional, while others are not. Hence, clustering all these meanings and classifying them as either compositional or idiomatic may be misleading. Instead, VPC and non-VPC uses of one specific verb-particle combination could be truly distinguished on the basis of frequency data, or, on the other hand, a word sense disambiguation approach may give an account of the compositional or idiomatic uses of the specific unit.

In our experiments, we use the Wiki50 corpus, in which VPCs are annotated in raw text, but no semantic classes are further distinguished. Hence, our goal here is not the automatic semantic classification of VPCs because we believe that first the identification of VPCs in context should be solved and then in a further step, genuine VPCs might be classified as compositional or idiomatic, given a manually annotated dataset from which this kind of information may be learnt. This issue will be addressed in a future study.

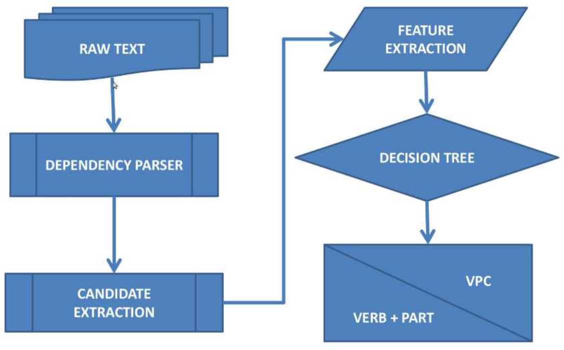

Figure 2: System Architecture

4 VPC Detection

Our goal is to identify each individual VPC in running texts; i.e. to take individual inputs like *How did they get on yesterday?* and mark each VPC in the sentence. Our tool called VPCTagger is based on a two-step approach. First, we syntactically parse each sentence, and extract potential VPCs with a syntax-based candidate extraction method. Afterwards, a binary classification can be used to automatically classify potential VPCs as VPCs or not. For the automatic classification of candidate VPCs, we implemented a machine learning approach, which is based on a rich feature set with new features like semantic and contextual features. Figure 2 outlines the process used to identify each individual VPC in a running text.

4.1 Corpora

To evaluate of our methods, we made use of two corpora. Statistical data on the corpora can be seen in Table 1. First, we used Wiki50 (Vincze et al., 2011), in which several types of multiword expressions (including VPCs) and Named Entities were marked. This corpus consists of 50 Wikipedia pages, and contains 466 occurrences of VPCs.

Corpus	Sentences	Tokens	VPCs	#
Wiki50	4,350	114,570	466	342
Tu&Roth	1,348	38,132	878	23

Table 1: Statistical data on the corpora.

In order to compare the performance of our system with others, we also used the dataset of Tu and Roth (2012), which contains 1,348 sentences taken from different parts of the British National Corpus. However, they only focused on VPCs in this dataset, where 65% of the sentences contain a phrasal verb and 35% contain a simplex verb-preposition combination. As Table 1 indicates, the Tu&Roth dataset only focused on 23 different VPCs, but 342 unique VPCs were annotated in the Wiki50 corpus.

4.2 Candidate Extraction

In this section, we concentrate on the first step of our approach, namely how VPC candidates can be selected from texts. As we mentioned in Section 1, our hypothesis is that the automatic detection of VPCs can be basically carried out by dependency parsers. Thus, we examined the performance of two parsers on VPC-specific syntactic labels.

As we had a full-coverage VPC annotated corpus where each individual occurrence of a VPC was manually marked, we were able to examine the characteristics of VPCs in a running text and evaluate the effectiveness of the parsers on this task. Therefore, here we examine dependency relations among the manually annotated gold standard VPCs, provided by the Stanford parser (Klein and Manning, 2003) and the Bohnet parser (Bohnet, 2010) for the Wiki50 corpus. In order to compare the efficiency of the parsers, both were applied using the same dependency representation. We found that only 52.57% and 58.16% of the annotated VPCs in Wiki50 had a verb-particle syntactic relation when we used the Stanford and Bohnet parsers, respectively. As Table 2 shows, there are several other syntactic constructions in which VPCs may occur.

Edge type	**Stanford**		**Bohnet**	
	#	%	#	%
prt	235	52.57	260	58.16
prep	23	5.15	107	23.94
advmod	56	12.52	64	14.32
sum	314	70.24	431	96.42
other	8	1.79	1	0.22
none	125	27.97	15	3.36
sum	447	100.00	447	100.00

Table 2: Edge types in the Wiki50 corpus. prt: particle. prep: preposition. advmod: adverbial modifier. other: other dependency labels. none: no direct syntactic connection between the verb and particle.

Therefore, we extended our candidate extraction method, where besides the *verb-particle* dependency relation, the *preposition* and *adverbial modifier* syntactic relations were also investigated among verbs and particles. With this modification, 70.24% and 96.42% of VPCs in the Wiki50 corpus could be identified. In this phase, we found that the Bohnet parser was more successful on the Wiki50 corpus, i.e. it could cover more VPCs, hence we applied the Bohnet parser in our further experiments.

Some researchers filtered LVC candidates by selecting only certain verbs that may be part of the construction. One example is Tu and Roth (2012), where the authors examined a verb-particle combination only if the verbal components were formed with one of the previously given six verbs (i.e. *make*, *take*, *have*, *give*, *do*, *get*).

Since Wiki50 was annotated for all VPC occurrences, we were able to check what percentage of VPCs could be covered if we applied this selection. As Table 3 shows, the six verbs used by Tu and Roth (2012) are responsible for only 50 VPCs on the Wiki50 corpus, so it covers only 11.16% of all gold standard VPCs.

Table 4 lists the most frequent VPCs and the verbal components on the Wiki50 corpus. As can be seen, the top 10 VPCs are responsible for only 17.41% of the VPC occurrences, while the top 10 verbal components are responsible for 41.07% of the VPC occurrences in the Wiki50 corpus. Furthermore, 127 different verbal component occurred in Wiki50, but the verbs *have* and *do* – which are used by Tu and Roth (2012) – do not appear in the corpus as verbal component of VPCs. All this indicates that applying lexical restrictions and focusing on a reduced set of verbs will lead to the exclusion of a considerable number of VPCs occurring in free texts and so, real-world tasks would hardly profit from them.

verb	**#**
take	27
get	10
give	5
make	3
have	0
do	0
sum	**50**

Table 3: The frequency of verbs on the Wiki50 corpus used by Tu and Roth (2012).

VPC	#	verb	#
call for	11	set	28
point out	9	take	27
carry out	9	turn	26
set out	8	go	21
grow up	8	call	21
set up	7	come	15
catch up	7	carry	13
turn on	7	look	13
take up	6	break	10
pass on	6	move	10
sum	**78**	**sum**	**184**

Table 4: The most frequent VPCs and verbal components on the Wiki50 corpus.

4.3 Machine Learning Based Candidate Classication

In order to perform an automatic classification of the candidate VPCs, a machine learning-based approach was implemented, which will be elaborated upon below. This method is based on a rich feature set with the following categories: orthographic, lexical, syntactic, and semantic. Moreover, as VPCs are highly ambiguous in raw texts, contextual features are also required.

- Orthographic features: Here, we examined whether the candidate **consists of two or more tokens**. Moreover, if the particle component **started with 'a'**, which prefix, in many cases, etymologically denotes a movement (like *across* and *away*), it was also noted and applied as a feature.

- Lexical features: We exploited the fact that the **most common verbs** occur most frequently in VPCs, so we selected fifteen verbs from the most frequent English verbs [1]. Here, we examined whether the lemmatised verbal component of the candidate was one of these fifteen verbs. We also examined whether the particle component of the potential VPC occurred among the **common English particles**. Here, we apply a manually built particle list based on linguistic considerations. Moreover, we also checked whether a potential VPC is contained in the **list of typical English VPCs** collected by Baldwin (2008).

- Syntactic features: the **dependency label** between the verb and the particle can also be exploited in identifying LVCs. As we typically found when dependency parsing the corpus, the syntactic relation between the verb and the particle in a VPC is `prt`, `prep` or `advmod` – applying the Stanford parser dependency representation, hence these syntactic relations were defined as features. If the candidate's **object was a personal pronoun**, it was also encoded as another syntactic feature.

- Semantic features: These features were based on the fact that the meaning of VPCs may typically reflect a motion or location like *go on* or *take away*. First, we examine that the verbal component is a **motion verb** like *go* or *turn*, or the **particle indicates a direction** like *out* or *away*.

  Moreover, the **semantic type of the prepositional object, object and subject** in the sentence can also help to decide whether the candidate is a VPC or not. Consequently, the `person`, `activity`, `animal`, `artifact` and `concept` semantic senses were looked for among the upper level hyperonyms of the nominal head of the prepositional object, object and subject in Princeton WordNet 3.1[2].

When several different machine learning algorithms were experimented on this feature set, the preliminary results showed that decision trees performed the best on this task. This is probably due to the fact that our feature set consists of a few compact (i.e. high-level) features. The J48 classifier of the WEKA package (Hall et al., 2009) was trained with its default settings on the above-mentioned feature set, which implements the C4.5 (Quinlan, 1993) decision tree algorithm. Moreover, Support Vector Machines (SVM) (Cortes and Vapnik, 1995) results are also reported to compare the performance of our methods with that of Tu and Roth (2012).

As the investigated corpora were not sufficiently large for splitting them into training and test sets of appropriate size, we evaluated our models in a cross validation manner on the Wiki50 corpus and the Tu&Roth dataset.

[1]http://en.wikipedia.org/wiki/Most_common_words_in_English

[2]http://wordnetweb.princeton.edu/perl/webwn

As Tu and Roth (2012) presented only the accuracy scores on the Tu & Roth dataset, we also employed an accuracy score as an evaluation metric on this dataset, where positive and negative examples were also marked. But, in the case of Wiki50 corpus, where only the positive VPCs were manually annotated, the $F_{\beta=1}$ score was employed and interpreted on the positive class as an evaluation metric. Moreover, all potential VPCs were treated as negative that were extracted by the candidate extraction method but were not marked as positive in the gold standard. Thus, in the resulting dataset negative examples are over-represented.

As Table 2 shows, the candidate extraction method did not cover all manually annotated VPCs in the Wiki50 corpus. Hence, we treated the omitted LVCs as false negatives in our evaluation.

As a baseline, we applied a context-free dictionary lookup method. In this case, we applied the same VPC list that was described among the lexical features. Then we marked candidates of the syntax-based method as VPC if the candidate VPC was found in the list. We also compared our results with the rule-based results available for Wiki50 (Nagy T. and Vincze, 2011) and also with the 5-fold cross validation results of Tu and Roth (2012).

5 Results

Table 5 lists the results obtained using the baseline dictionary lookup, rule-based method, dependency parsers and machine learning approaches on the Wiki50 corpus. It is revealed that the dictionary lookup method performed worst and achieved an F-score of 35.43. Moreover, this method only achieved a precision score of 49.77%. However, the rule-based method achieved the highest precision score with 91.26%, but the dependency parsers also got high precision scores of about 90% on Wiki50. It is also clear that the machine learning-based approach, the VPCTagger, is the most successful method on Wiki50: it achieved an F-score 10 points higher than those for the rule-based method and dependency parsers and more than 45 points higher than that for the dictionary lookup.

In order to compare the performance of our system with others, we evaluated it on the Tu&Roth dataset (Tu and Roth, 2012). Table 6 compares the results achieved by the dictionary lookup and the rule-based method on the Tu&Roth dataset. Moreover, it also lists the results of Tu and Roth (2012) and the VPCTagger evaluated in the 5-fold cross validation manner, as Tu and Roth (2012) applied this evaluation schema. As in the Tu&Roth dataset positive and negative examples were also marked, we were able to use accuracy as evaluation metric besides the $F_{\beta=1}$ scores. It is revealed that the dictionary lookup and the rule-based method achieved an F-score of about 50, but our method seems the most successful on this dataset, as it can yield an accuracy 3.32% higher than that for the Tu&Roth system.

Method	Prec.	Rec.	F-score
Dictionary Lookup	49.77	27.5	35.43
Rule-based	**91.26**	58.52	71.31
Stanford Parser	91.09	52.57	66.67
Bohnet Parser	89.04	58.16	70.36
ML J48	85.7	**76.79**	**81.0**
ML SVM	89.07	65.62	75.57

Table 5: Results obtained in terms of precision, recall and F-score.

Method	Accuracy	F-score
Dictionary Lookup	51.13	52.24
Rule Based	56.92	43.84
VPCTagger	**81.92**	**85.69**
Tu&Roth	78.6%	–

Table 6: 5-fold cross validation results on the Tu&Roth dataset in terms of accuracy and F-score.

6 Discussion

The applied machine learning-based method extensively outperformed our dictionary lookup and rule-based baseline methods, which underlines the fact that our approach can be suitably applied to VPC detection in raw texts. It is well demonstrated that VPCs are very ambiguous in raw text, as the dictionary lookup method only achieved a precision score of 49.77% on the Wiki50 corpus. This demonstrates that the automatic detection of VPCs is a challenging task and contextual features are essential. In the case of the dictionary lookup, to achieve a higher recall score was mainly limited by the size of the dictionary used.

As Table 5 shows, VPCTagger achieved an F-score 10% higher than those for the dependency parsers, which may refer to the fact that our machine learning-based approach performed well on this task. This method proved to be the most balanced as it got roughly the same recall, precision and F-score results on the Wiki50 corpus. In addition, the dependency parsers achieve high precision with lower recall scores.

Moreover, the results obtained with our machine learning approach on the Tu&Roth dataset outperformed those reported in Tu and Roth (2012). This may be attributed to the inclusion of a rich feature set with new features like semantic and contextual features that were used in our system.

As Table 6 indicates, the dictionary lookup and rule-based methods were less effective when applied on the Tu&Roth dataset. Since the corpus was created by collecting sentences that contained phrasal verbs with specific verbs, this dataset contains a lot of negative and ambiguous examples besides annotated VPCs, hence the distribution of VPCs in the Tu&Roth dataset is not comparable to those in Wiki50, where each occurrence of a VPCs were manually annotated in a running text. Moreover, in this dataset, only one positive or negative example was annotated in each sentence, and they examined just the verb-particle pairs formed with the six verbs as a potential VPC. However, the corpus probably contains other VPCs which were not annotated. For example, in the sentence *The agency* ***takes on*** *any kind of job – you just name the subject and give us some indication of the kind of thing you want to know, and then we* ***go out*** *and* ***get it for*** *you.*, the only phrase *takes on* was listed as a positive example in the Tu&Roth dataset. But two examples, (*go out* – positive and *get it for* – negative) were not marked. This is problematic if we would like to evaluate our candidate extractor on this dataset as it would identify all these phrases, even if it is restricted to verb-particle pairs containing one of the six verbs mentioned above, thus yielding false positives already in the candidate extraction phase.

In addition, this dataset contains 878 positive VPC occurrences, but only 23 different VPCs. Consequently, some positive examples were overrepresented. But the Wiki50 corpus may contain some rare examples and it probably reflects a more realistic distribution as it contains 342 unique VPCs.

A striking difference between the Tu & Roth database and Wiki50 is that while Tu and Roth (2012) included the verbs *do* and *have* in their data, they do not occur at all among the VPCs collected from Wiki50. Moreover, these verbs are just responsible for 25 positive VPCs examples in the Tu & Roth dataset. Although these verbs are very frequent in language use, they do not seem to occur among the most frequent verbal components concerning VPCs. A possible reason for this might be that VPCs usually contain a verb referring to movement in its original sense and neither *have* nor *do* belong to motion verbs.

An ablation analysis was carried out to examine the effectiveness of each individual feature types of the machine learning based candidate classification. Besides the feature classification described in Section 4.3, we also examined the effectiveness of the contextual features. In this case, the feature which examined whether the candidates object was a personal pronoun or not and the semantic type of the prepositional object, object and subject were treated as contextual features. Table 7 shows the usefulness of each individual feature type on the Wiki50 corpus. For each feature type, a J48 classifier was trained with all of the features except that one. Then we compared the performance to that got with all the features. As the ablation analysis shows, each type of feature contributed to the overall performance. We found that the lexical and orthographic features were the most powerful, the semantic, syntactic features were also useful; while contextual features were less effective, but were still exploited by the model.

Features	**Prec.**	**Rec.**	**F-score**	Diff.
All	**85.7**	**76.79**	**81.0**	–
Semantic	86.55	66.52	75.22	-5.78
Orthographic	83.26	65.85	73.54	-7.46
Syntax	84.31	71.88	77.6	-3.4
Lexical	89.68	60.71	72.41	**-8.59**
Contextual	86.68	74.55	80.16	-0.84

Table 7: The usefulness of individual features in terms of precision, recall and F-score using the Wiki50 corpus.

The most important features in our system are lexical ones, namely, the lists of the most frequent English verbs and particles. It is probably due to the fact that the set of verbs used in VPCs is rather limited, furthermore, particles form a closed word class that is, they can be fully listed, hence the particle component of a VPC will necessarily come from a well-defined set of words.

Besides the ablation analysis, we also investigated the decision tree model produced by our experiments. The model profited most from the syntactic and lexical features, i.e. the dependency label provided by the parsers between the verb and the particle also played an important role in the classification process.

We carried out a manual error analysis in order to find the most typical errors our system made. Most errors could be traced back to POS-tagging or parsing errors, where the particle was classified as a preposition. VPCs that include an adverb (as labeled by the POS tagger and the parser) were also somewhat more difficult to identify, like *come across* or *go back*. Preposition stranding (in e.g. relative clauses) also resulted in false positives like in *planets he had an adventure on*.

Other types of multiword expressions were also responsible for errors. For instance, the system classified *come out* as a VPC within the idiom *come out of the closet* but the gold standard annotation in Wiki50 just labeled the phrase as an idiom and no internal structure for it was marked. A similar error could be found for light verb constructions, for example, *run for office* was marked as a VPC in the data, but *run for* was classified as a VPC, yielding a false positive case. Multiword prepositions like *up to* also led to problems: in *he taught up to 1986*, *taught up* was erroneously labeled as VPC. Finally, in some cases, annotation errors in the gold standard data were the source of mislabeled candidates.

7 Conclusions

In this paper, we focused on the automatic detection of verb-particle combinations in raw texts. Our hypothesis was that parsers trained on texts annotated with extra information for VPCs can identify VPCs in texts. We introduced our machine learning-based tool called VPCTagger, which allowed us to automatically detect VPCs in context. We solved the problem in a two-step approach. In the first step, we extracted potential VPCs from a running text with a syntax-based candidate extraction method and we applied a machine learning-based approach that made use of a rich feature set to classify extracted syntactic phrases in the second step. In order to achieve a greater efficiency, we defined several new features like semantic and contextual, but according to our ablation analysis we found that each type of features contributed to the overall performance.

Moreover, we also examined how syntactic parsers performed in the VPC detection task on the Wiki50 corpus. Furthermore, we compared our methods with others when we evaluated our approach on the Tu&Roth dataset. Our method yielded better results than those got using the dependency parsers on the Wiki50 corpus and the method reported in (Tu and Roth, 2012) on the Tu&Roth dataset.

Here, we also showed how dependency parsers performed on identifying VPCs, and our results indicate that although the dependency label provided by the parsers is an essential feature in determining whether a specific VPC candidate is a genuine VPC or not, the results can be further improved by extending the system with additional features like lexical and semantic features. Thus, one possible application of the VPCTagger may be to help dependency parsers: based on the output of VPCTagger, syntactic labels provided by the parsers can be overwritten. With backtracking, the accuracy of syntactic parsers may increase, which can be useful for a number of higher-level NLP applications that exploit syntactic information.

In the future, we would like to improve our system by defining more complex contextual features. We also plan to examine how the VPCTagger improve the performance of higher level NLP applications like machine translation systems, and we would also like to investigate the systematic differences among the performances of the parsers and VPCTagger, in order to improve the accuracy of parsing. In addition, we would like to compare different automatic detection methods of multiword expressions, as different types of MWEs are manually annotated in the Wiki50 corpus.

Acknowledgments

István Nagy T. was partially funded by the National Excellence Program TÁMOP-4.2.4.A/2-11/1-2012-0001 of the State of Hungary, co-financed by the European Social Fund. Veronika Vincze was funded in part by the European Union and the European Social Fund through the project FuturICT.hu (grant no.: TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

References

Timothy Baldwin and Aline Villavicencio. 2002. Extracting the unextractable: A case study on verb-particles. In *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Timothy Baldwin. 2005. Deep lexical acquisition of verb-particle constructions. *Computer Speech and Language*, 19(4):398–414, October.

Timothy Baldwin. 2008. A resource for evaluating the deep lexical acquisition of English verb-particle constructions. In *Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions (MWE 2008)*, pages 1–2.

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of Coling 2010*, pages 89–97.

Corinna Cortes and Vladimir Vapnik. 1995. *Support-vector networks*, volume 20. Kluwer Academic Publishers.

Spence Green, Marie-Catherine de Marneffe, and Christopher D. Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics*, 39(1):195–227.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18.

Ray Jackendoff. 2002. English particle constructions, the lexicon, and the autonomy of syntax. In Nicole Deh, Ray Jackendoff, Andrew McIntyre, and Silke Urban, editors, *Verb-Particle Explorations*, pages 67–94, Berlin / New York. Mouton de Gruyter.

Su Nam Kim and Timothy Baldwin. 2006. Automatic identification of English verb particle constructions using linguistic features. In *Proceedings of the Third ACL-SIGSEM Workshop on Prepositions*, pages 65–72.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Annual Meeting of the ACL*, volume 41, pages 423–430.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–331.

Francesca Masini. 2005. Multi-word expressions between syntax and the lexicon: The case of Italian verb-particle constructions. *SKY Journal of Linguistics*, 18:145–173.

Diana McCarthy, Bill Keller, and John Carroll. 2003. Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment - Volume 18*, MWE '03, pages 73–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

István Nagy T. and Veronika Vincze. 2011. Identifying Verbal Collocations in Wikipedia Articles. In *Proceedings of the 14th International Conference on Text, Speech and Dialogue*, TSD'11, pages 179–186, Berlin, Heidelberg. Springer-Verlag.

Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In *Proceedings of CICLing 2002*, pages 1–15, Mexico City, Mexico.

Yuancheng Tu and Dan Roth. 2012. Sorting out the Most Confusing English Phrasal Verbs. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pages 65–69, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aline Villavicencio. 2003. Verb-particle constructions and lexical resources. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment - Volume 18*, MWE '03, pages 57–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

Veronika Vincze, István Nagy T., and Gábor Berend. 2011. Multiword Expressions and Named Entities in the Wiki50 Corpus. In *Proceedings of RANLP 2011*, pages 289–295, Hissar, Bulgaria, September. RANLP 2011 Organising Committee.

Veronika Vincze, János Zsibrita, and István Nagy T. 2013. Dependency Parsing for Identifying Hungarian Light Verb Constructions. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 207–215, Nagoya, Japan, October. Asian Federation of Natural Language Processing.