Extracting the Unextractable: A Case Study on Verb-particles

Timothy Baldwin* **and Aline Villavicencio†**

* CSLI, Ventura Hall, Stanford University
Stanford, CA 94305-4115 USA
tbaldwin@csli.stanford.edu

† University of Cambridge, Computer Laboratory, William Gates Building
JJ Thomson Avenue, Cambridge CB3 OFD, UK
Aline.Villavicencio@cl.cam.ac.uk

Abstract

This paper proposes a series of techniques for extracting English verb–particle constructions from raw text corpora. We initially propose three basic methods, based on tagger output, chunker output and a chunk grammar, respectively, with the chunk grammar method optionally combining with an attachment resolution module to determine the syntactic structure of verb–preposition pairs in ambiguous constructs. We then combine the three methods together into a single classifier, and add in a number of extra lexical and frequentistic features, producing a final F-score of 0.865 over the WSJ.

1 Introduction

There is growing awareness of the pervasiveness and idiosyncrasy of **multiword expressions** (MWEs), and the need for a robust, structured handling thereof (Sag et al., 2002; Calzolari et al., 2002; Copestake et al., 2002). Examples of MWEs are lexically fixed expressions (e.g. *ad hoc*), idioms (e.g. *see double*), light verb constructions (e.g. *make a mistake*) and institutionalised phrases (e.g. *kindle excitement*).

MWEs pose a challenge to NLP due to their syntactic and semantic idiosyncrasies, which are often unpredictable from their component parts. Large-scale manual annotation of MWEs is infeasible due to their sheer volume (at least equivalent to the number of simplex words (Jackendoff, 1997)), productivity and domain-specificity. Ideally, therefore, we would like to have some means of automatically extracting MWEs from a given domain or corpus, allowing us to pre-tune our grammar prior to deployment. It is this task of extraction that we target in this paper. This research represents a component of the LinGO multiword expression project,[1] which is targeted at extracting, adequately handling and representing MWEs of all types. As a research testbed and target resource to expand/domain-tune, we use the LinGO English Resource Grammar (LinGO-ERG), a linguistically-precise HPSG-based grammar under development at CSLI (Copestake and Flickinger, 2000; Flickinger, 2000).

The particular MWE type we target for extraction is the English **verb-particle construction.** Verb-particle constructions ("VPCs") consist of a head verb and one or more obligatory **particles**, in the form of intransitive prepositions (e.g. *hand in*), adjectives (e.g. *cut short*) or verbs (e.g. *let go*) (Villavicencio and Copestake, 2002a; Villavicencio and Copestake, 2002b; Huddleston and Pullum, 2002); for the purposes of this paper, we will focus exclusively on prepositional particles—by far the most common and productive of the three types—and further restrict our attention to single-particle VPCs (i.e. we ignore VPCs such as *get along together*). We define VPCs to optionally select for an NP complement, i.e. to occur both transitively (e.g. *hand in the paper*) and intransitively (e.g. *battle on*).

One aspect of VPCs that makes them difficult to extract (cited in, e.g., Smadja (1993)) is that the verb and particle can be non-contiguous, e.g. *hand the paper in* and *battle right on*. This sets them apart from conventional collocations and terminology (see, e.g., Manning and Schütze (1999) and McKeown and Radev (2000)) in that they cannot be captured effectively using N-grams, due to the variability in the number and type of words potentially interceding between the verb and particle.

We are aiming for an extraction technique which is applicable to any raw text corpus, allowing us to tune grammars to novel domains. Any linguistic annotation required during the extraction process, therefore, is produced through automatic means, and it is only for reasons of accessibility and comparability with other research that we choose to work over the Wall Street Journal section of the Penn Treebank (Marcus et al., 1993). That is, other than in establishing upper bounds on the performance of the different extraction methods, we use only the raw text component of the treebank.

In this paper, we first outline distinguishing features of VPCs relevant to the extraction process (§ 2). We then present and evaluate a number of simple methods for extracting VPCs based on, respectively, POS tagging (§ 3), the output of a full text chunk parser (§ 4), and a chunk grammar (§ 5). Finally, we detail enhancements to the basic methods (§ 6) and give a brief description of related research (§ 7) before concluding the paper (§ 8).

2 Distinguishing Features of VPCs

Here, we review a number of features of VPCs pertinent to the extraction task. First, we describe linguistic qualities that characterise VPCs, and second

[1] http://lingo.stanford.edu/mwe

we analyse the actual occurrence of VPCs in the WSJ.

2.1 Linguistic features

Given an arbitrary verb–preposition pair, where the preposition is governed by the verb, a number of analyses are possible. If the preposition is intransitive, a VPC (either intransitive or transitive) results. If the preposition is transitive, it must select for an NP, producing either a **prepositional verb** (e.g. *refer to*) or a **free verb–preposition combination** (e.g. *put it on the table*, *climb up the ladder*).

A number of diagnostics can be used to distinguish VPCs from both prepositional verbs and free verb–preposition combinations (Huddleston and Pullum, 2002):

1. transitive VPCs undergo the particle alternation
2. with transitive VPCs, pronominal objects must be expressed in the "split" configuration
3. manner adverbs cannot occur between the verb and particle

The first two diagnostics are restricted to transitive VPCs, while the third applies to both intransitive and transitive VPCs.

The first diagnostic is the canonical test for particlehood, and states that transitive VPCs take two word orders: the **joined** configuration whereby the verb and particle are adjacent and the NP complement follows the particle (e.g. *hand in the paper*), and the **split** configuration whereby the NP complement occurs between the verb and particle (e.g. *hand the paper in*). Note that prepositional verbs and free verb–preposition combinations can occur only in the joined configuration (e.g. *refer to the book* vs. **refer the book to*). Therefore, the existence of a verb–preposition pair in the split configuration is sufficient evidence for a VPC analysis. It is important to realise that compatibility with the particle alternation is a sufficient but not necessary condition on verb–particlehood. That is, a small number of VPCs do not readily occur in the split configuration, including *carry out (a threat)* (cf. *?carry a threat out*).

The second diagnostic stipulates that pronominal NPs can occur only in the split configuration (*hand it in* vs. **hand in it*). Note also that heavy NPs tend to occur in the joined configuration, and that various other factors interact to determine which configuration a given VPC in context will occur in (see, e.g., Gries (2000)).

The third diagnostic states that manner adverbs cannot intercede between the verb and particle (e.g. **hand quickly the paper in*). Note that this constraint is restricted to manner adverbs, and that there is a small set of adverbs which can pre-modify particles and hence occur between the verb and particle (e.g. *well* in *jump well up*).

2.2 Corpus occurrence

In order to get a feel for the relative frequency of VPCs in the corpus targeted for extraction, namely the WSJ section of the Penn Treebank, we took a random sample of 200 VPCs from the Alvey Natural Language Tools grammar (Grover et al., 1993) and did a manual corpus search for each. In the case that a VPC was found attested in the WSJ, we made a note of the frequency of occurrence as: (a) an intransitive VPC, (b) a transitive VPC in the joined configuration, and (c) a transitive VPC in the split configuration. Of the 200 VPCs, only 62 were attested in the Wall Street Journal corpus (WSJ), at a mean token frequency of 5.1 and median token frequency of 2 (frequencies totalled over all 3 usages). Figure 1 indicates the relative proportion of the 62 attested VPC types which occur with the indicated frequencies. From this, it is apparent that two-thirds of VPCs occur at most three times in the overall corpus, meaning that any extraction method must be able to handle extremely sparse data.

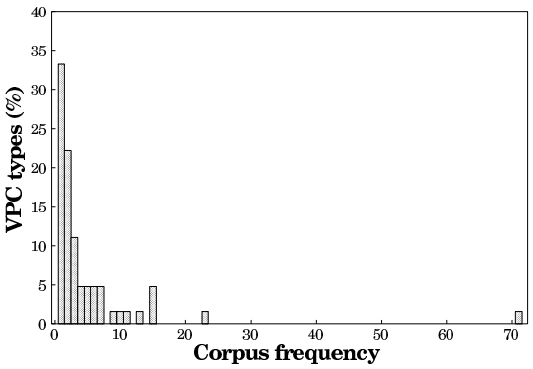

Figure 1: Frequency distribution of VPCs in the WSJ

Tagger	$\frac{correct}{extracted}$	*Prec*	*Rec*	$F_{\beta=1}$
Brill	$\frac{135}{135}$	1.000	0.177	0.301
Penn	$\frac{667}{800}$	0.834	0.565	0.673

Table 1: POS-based extraction results

Of the 62 attested VPCs, 29 have intransitive usages and 45 have transitive usages. Of the 45 attested transitive VPCs, 12 occur in both the joined and split configurations and can hence be unambiguously identified as VPCs based on the first diagnostic from above. For the remaining 33 transitive VPCs, we have only the joined usage, and must find some alternate means of ruling out a prepositional verb or free verb–preposition combination analysis. Note that for the split VPCs, the mean number of words occurring between the verb and particle was 1.6 and the maximum 3.

In the evaluation of the various extraction techniques below, recall is determined relative to this limited set of 62 VPCs attested in the WSJ. That is, recall is an indication of the proportion of the 62 VPCs contained within the set of extracted VPCs.

3 Method-1: Simple POS-based Extraction

One obvious method for extracting VPCs is to run a simple regular expression over the output of a part-of-speech (POS) tagger, based on the observation that the Penn Treebank POS tagset, e.g., contains a dedicated particle tag (`RP`). Given that all particles are governed by a verb, extraction consists of simply locating each particle and searching back (to the left of the particle, as particles cannot be passivised or otherwise extraposed) for the head verb of the VPC. Here and for the subsequent methods, we assume that the maximum word length for NP complements in the split configuration for transitive VPCs is 5,[2] i.e. that an NP "heavier" than this would occur more naturally in the joined configuration. We thus discount all particles which are more than 5 words from their governing verb. Additionally, we extracted a set of 73 canonical particles from the LinGO-ERG, and used this to filter out extraneous particles in the POS data.

In line with our assumption of raw text to extract over, we use the Brill tagger (Brill, 1995) to automatically tag the WSJ, rather than making use of the manual POS annotation provided in the Penn Treebank. We further lemmatise the data using morph (Minnen et al., 2001) and extract VPCs based on the Brill tags. This produces a total of 135 VPCs, which we evaluate according to the standard metrics of precision (*Prec*), recall (*Rec*) and F-score ($F_{\beta=1}$). Note that here and for the remainder of this paper, precision is calculated according to the manual annotation for the combined total of 4,173 VPC candidate types extracted by the various methods described in this paper, whereas recall is relative to the 62 attested VPCs from the Alvey Tools data as described above.

As indicated in the first line of Table 1 ("Brill"), the simple POS-based method results in a precision of 1.000, recall of 0.177 and F-score of 0.301.

In order to determine the upper bound on performance for this method, we ran the extraction method over the original tagging from the Penn Treebank. This resulted in an F-score of 0.774 ("Penn" in Table 1). The primary reason for the large disparity between the Brill tagger output and original Penn Treebank annotation is that it is notoriously difficult to differentiate between particles, prepositions and adverbs (Toutanova and Manning, 2000). Over the WSJ, the Brill tagger achieves a modest tag recall of 0.103 for particles, and tag precision of 0.838. That is, it is highly conservative in allocating particle tags, to the extent that it recognises only two particle types for the whole of the WSJ: *out* and *down*.

4 Method-2: Simple Chunk-based Extraction

To overcome the shortcomings of the Brill tagger in identifying particles, we next look to full chunk parsing. Full chunk parsing involves partitioning up a text into syntactically-cohesive, head-final segments ("chunks"), without attempting to resolve inter-chunk dependencies. In the chunk inventory devised for the CoNLL-2000 test chunking shared task (Tjong Kim Sang and Buchholz, 2000), a dedicated particle chunk type once again exists. It is therefore possible to adopt an analogous approach to that from Method-1, in identifying particle chunks then working back to locate the verb each particle chunk is associated with.

[2]Note, this is the same as the maximum span length of 5 used by Smadja (1993), and above the maximum attested NP length of 3 from our corpus study (see Section 2.2).

WSJ			*CoNLL*		
Prec	*Rec*	$F_{\beta=1}$	*Prec*	*Rec*	$F_{\beta=1}$
0.889	0.911	0.900	0.912	0.925	0.919

Table 2: Chunking performance

4.1 Chunk parsing method

In order to chunk parse the WSJ, we first tagged the full WSJ and Brown corpora using the Brill tagger, and then converted them into chunks based on the original Penn Treebank parse trees, with the aid of the conversion script used in preparing the CoNLL-2000 shared task data.[3] We next lemmatised the data using morph (Minnen et al., 2000), and chunk parsed the WSJ with TiMBL 4.1 (Daelemans et al., 2001) using the Brown corpus as training data. TiMBL is a memory-based classification system based on the k-nearest neighbour algorithm, which takes as training data a set of fixed-length feature vectors pre-classified according to an information field. For each test instance described over the same feature vector, it then returns the "neighbours" at the k-nearest distances to the test instance and classifies the test instance according to the class distribution over those neighbours. TiMBL provides powerful functionality for determining the relative distance between different values of a given feature in the form of MVDM, and also supports weighted voting between neighbours in classifying inputs, e.g. in the form of inverse distance weighting.

We ran TiMBL based on the feature set described in Veenstra and van den Bosch (2000), that is using the 5 word lemmata and POS tags to the left and 3 word lemmata and POS tags to the right of each focus word, along with the POS tag and lemma for the focus word. We set k to 5, ran MVDM over only the POS tags[4] and used inverse distance weighting, but otherwise ran TiMBL with the default settings.

We evaluated the basic TiMBL method over both the full WSJ data, training on the Brown section of the Penn Treebank, and over the original shared task data from CoNLL-2000, the results for which are presented in Table 2. Note that, similarly to the CoNLL-2000 shared task, precision, recall and

[3]Note that the gold standard chunk data for the WSJ was used only in evaluation of chunking performance, and to establish upper bounds on the performance of the various extraction methods.

[4]Based on the results of Veenstra and van den Bosch (2000) and the observation that MVDM is temperamental over sparse data (i.e. word lemmata).

Chunker	$\frac{correct}{extracted}$	*Prec*	*Rec*	$F_{\beta=1}$
TiMBL	$\frac{695}{854}$	0.772	0.548	0.641
Penn	$\frac{651}{760}$	0.857	0.694	0.766

Table 3: Chunk tag-based extraction results

F-score are all evaluated at the *chunk* rather than the word level. The F-score of 0.919 for the CoNLL-2000 data is roughly the median score attained by systems performing in the original task, and slightly higher than the F-score of 0.915 reported by Veenstra and van den Bosch (2000), due to the use of word lemmata rather than surface forms, and also inverse distance weighting. The reason for the drop-off in performance between the CoNLL data and the full WSJ is due to the CoNLL training and test data coming from a homogeneous data source, namely a subsection of the WSJ, but the Brown corpus being used as the training data in chunking the full extent of the WSJ.

4.2 Extraction method

Having chunk-parsed the WSJ in the manner described above, we next set about extracting VPCs by identifying each particle chunk, and searching back for the governing verb. As for Method-1, we allow a maximum of 5 words to intercede between a particle and its governing verb, and we apply the additional stipulation that the only chunks that can occur between the verb and the particle are: (a) noun chunks, (b) preposition chunks adjoining noun chunks, and (c) adverb chunks found in our closed set of particle pre-modifiers (see § 2.1). Additionally, we used the gold standard set of 73 particles to filter out extraneous particle chunks, as for Method-1 above.

The results for chunk-based extraction are presented in Table 3, evaluated over the chunk parser output ("TiMBL") and also the gold-standard chunk data for the WSJ ("Penn"). These results are significantly better than those for Method-1 over the Brill output and Penn data, respectively, both in terms of the raw number of VPCs extracted and F-score. One reason for the relative success of extracting over chunker as compared to tagger output is that our chunker was considerably more successful than the Brill tagger at annotating particles, returning an F-score of 0.737 over particle chunks (precision=0.786, recall=0.693). The stipulations on particle type and what could occur between a verb and particle chunk were crucial in maintaining a high VPC extraction precision, relative to both particle chunk precision and the gold standard extraction precision. As can be seen from the upper bound on recall (i.e. recall over the gold standard chunk data), however, this method has limited applicability.

5 Method-3: Chunk Grammar-based Extraction

The principle weakness of Method-2 was recall, leading us to implement a rule-based chunk sequencer which searches for particles in prepositional and adverbial chunks as well as particle chunks. In essence,

Method	$\frac{correct}{extracted}$	*Prec*	*Rec*	$F_{\beta=1}$
RULE−ATT	$\frac{676}{1119}$	0.604	0.694	0.646
TIMBL−ATT	$\frac{615}{823}$	0.747	0.661	0.702
PENN−ATT	$\frac{694}{927}$	0.749	0.823	0.784
RULE+ATT	$\frac{951}{3126}$	0.304	0.823	0.444
TIMBL+ATT	$\frac{739}{1049}$	0.704	0.710	0.707
PENN+ATT	$\frac{750}{1079}$	0.695	0.871	0.773

Table 4: Chunk grammar-based extraction results

we take each verb chunk in turn, and search to the right for a single-word particle, prepositional or adverbial chunk which is contained in the gold standard set of 73 particles. For each such chunk pair, it then analyses: (a) the chunks which occur between them to ensure that, maximally, an NP and particle pre-modifier adverb chunk are found; (b) the chunks that occur immediately after the particle/preposition/adverb chunk to check for a clause boundary or NP; and (c) the clause context of the verb chunk for possible extraposition of an NP verbal complement, through passivisation or relativisation. The objective of this analysis is to both determine the valence of the VPC candidate (intransitive or transitive) and identify evidence either supporting or rejecting a VPC analysis. Evidence for or against a VPC analysis is in the form of congruence with the known linguistic properties of VPCs, as described in Section 2.1. For example, if a pronominal noun chunk were found to occur immediately after the (possibly) particle chunk (e.g. **see off him*), a VPC analysis would not be possible. Alternatively, if a punctuation mark (e.g. a full stop) were found to occur immediately after the "particle" chunk and nothing interceded between the verb and particle chunk, then this would be evidence for an intransitive VPC analysis.

The chunk sequencer is not able to furnish positive or negative evidence for a VPC analysis in all cases. Indeed, in a high proportion of instances, a noun chunk (=NP) was found to follow the "particle" chunk, leading to ambiguity between analysis as a VPC, prepositional verb or free verb–preposition combination (see Section 2.1), or in the case that an NP occurs between the verb and particle, the "particle" being the head of a PP post-modifying an NP. As a case in point, the VP *hand the paper in here* could take any of the following structures: (1) *hand [the paper] [in] [here]* (transitive VPC *hand in* with adjunct NP *here*), (2) *hand [the paper] [in here]* (transitive prepositional verb *hand in* or simple transitive verb with PP adjunct), and (3) *hand [the paper in here]* (simple transitive verb). In such cases, we can choose to either (a) avoid committing ourselves to any one analysis, and ignore all such ambiguous cases, or (b) use some means to resolve the attachment ambiguity (i.e. whether the NP is governed by the verb, resulting in a VPC, or the preposition, resulting in a prepositional verb or free verb–preposition combination). In the latter case,

we use an unsupervised attachment disambiguation method, based on the log-likelihood ratio ("LLR", Dunning (1993)). That is, we use the chunker output to enumerate all the verb–preposition, preposition–noun and verb–noun bigrams in the WSJ data, based on chunk heads rather than strict word bigrams. We then use frequency data to pre-calculate the LLR for each such type. In the case that the verb and "particle" are joined (i.e. no NP occurs between them), we simply compare the LLR of the verb–noun and particle–noun pairs, and assume a VPC analysis in the case that the former is strictly larger than the latter. In the case that the verb and "particle" are split (i.e. we have the chunk sequence VC NC_1 PC NC_2),[5] we calculate three scores: (1) the product of the LLR for (the heads of) VC-PC and VC-NC_2 (analysis as VPC, with NC_2 as an NP adjunct of the verb); (2) the product of the LLR for NC_1-PC and PC-NC_2 (transitive verb analysis, with the PP modifying NC_1); and (3) the product of the LLR for VC-PC and PC-NC_2 (analysis as prepositional verb or free verb–preposition combination). Only in the case that the first of these scores is strictly greater than the other two, do we favour a (transitive) VPC analysis.

Based on the positive and negative grammatical evidence from above, for both intransitive and transitive VPC analyses, we generate four frequency-based features. The optional advent of data derived through attachment resolution, again for both intransitive and transitive VPC analyses, provides another two features. These features can be combined in either of two ways: (1) in a rule-based fashion, where a given verb–preposition pair is extracted out as a VPC only in the case that there is positive and no negative evidence for either an intransitive or transitive VPC analysis ("RULE" in Table 4); and (2) according to a classifier, using TiMBL to train over the auto-chunked Brown data, with the same basic settings as for chunking (with the exception that each feature is numeric and MVDM is not used — results presented as "TIMBL" in Table 4). We also present upper bound results for the classifier-based method using gold standard chunk data, rather than the chunker output ("PENN"). For each of these three basic methods, we present results with and without the attachment-resolved data ("±ATT").

Based on the results in Table 4, the classifier-based method ("TIMBL") is superior to not only the rule-based method ("RULE"), but also Method-1 and Method-2. While the rule-based method degrades significantly when the attachment data is factored in, the classifier-based method remains at the same basic F-score value, undergoing a drop in precision but equivalent gain in recall and gaining more than 120 correct VPCs in the process. RULE+ATT returns the highest recall value of all the automatic methods to date at 0.823, at the cost of low precision at 0.304. This points to the attachment disambiguation method having high recall but low precision. TIMBL±ATT and PENN±ATT are equivalent in terms of precision, but the Penn data leads to considerably better recall.

[5]Here, VC = verb chunk, NC = noun chunk and PC = (intransitive or transitive) preposition chunk.

Method	$\frac{correct}{extracted}$	*Prec*	*Rec*	$F_{\beta=1}$
Combine	$\frac{719}{953}$	0.754	0.710	0.731
M_2^*	$\frac{686}{778}$	0.882	0.677	0.766
M_3−ATT*	$\frac{684}{788}$	0.868	0.694	0.771
M_3+ATT*	$\frac{871}{1020}$	0.854	0.823	0.838
Combine*	$\frac{1000}{1164}$	0.859	0.871	0.865
$Combine^*_{Penn}$	$\frac{931}{1047}$	0.889	0.903	0.896

Table 5: Consolidated extraction results

6 Improving on the Basic Methods

Comparing the results for the three basic methods, it is apparent that Method-1 and Method-2 offer higher precision while Method-3 offers higher recall. In order to capitalise on the respective strengths of the different methods, in this section, we investigate the possibility of combining the outputs of the four methods into a single consolidated classifier. System combination is achieved by taking the union of all VPC outputs from all systems, and having a vector of frequency-based features for each, based on the outputs of the different methods for the VPC in question. For each of Method-1 and Method-2, a single feature is used describing the total number of occurrences of the given VPC detected by that method. For Method-3, we retain the 6 features used as input to TIMBL±ATT, namely the frequency with which positive and negative evidence was detected and also the frequency of VPCs detected through attachment resolution, for both intransitive and transitive VPCs. Training data comes from the output of the different methods over the Brown corpus, and the chunking data for Method-2 and Method-3 was generated using the WSJ gold standard chunk data as training data, analogously to the method used to chunk parse the WSJ.

The result of this simple combination process is presented in the first line of Table 5 ("Combine"). Encouragingly, we achieved the exact same recall as the best of the simple methods (TIMBL+ATT) at 0.710, and significantly higher F-score than any individual method at 0.731.

Steeled by this initial success, we further augment the feature space with features describing the frequency of occurrence of: (a) the particle in the corpus, and (b) deverbal noun and adjective forms of the VPC in the corpus (e.g. *turnaround*, *dried-up*), determined through a simple concatenation operation optionally inserting a hyphen. The first of these is attempted to reflect the fact that high-frequency particles (e.g. *up*, *over*) are more productive (i.e. are found in novel VPCs more readily) than low-frequency particles.[6] The deverbal feature is intended to reflect the fact that VPCs have the po-

[6]We also experimented with a similar feature describing verb frequency, but found it to either degrade or have no effect on classifier performance.

tential to undergo deverbalisation whereas prepositional verbs and free verb–preposition combinations do not.[7] We additionally added in features describing: (a) the number of letters in the verb lemma, (b) the verb lemma, and (c) the particle lemma. The first feature was intended to capture the informal observation that shorter verbs tend to be more productive than longer verbs (which offers one possible explanation for the anomalous *call/ring/phone/*telephone up*). The second and third features are intended to capture this same productivity effect, but on a individual word-level. Note that as TiMBL treats all features as fully independent, it is not able to directly pick up on the gold standard verb–particle pairs in the training data to select in the test data.

The expanded set of features was used to re-evaluate each of: Method-2 (M_2^* in Table 5); the classifier version of Method-3 with and without attachment-resolved data ($M_3 \pm ATT^*$); and the simple system combination method (Combine*). Additionally, we calculated an upper bound for the expanded feature set based on the gold standard data for each of the methods ($Combine^*_{Penn}$ in Table 5). The results for these five consolidated methods are presented in Table 5.

The addition of the 7 new features leads to an appreciable gain in both precision and recall for all methods, with the system combination method once again proving to be the best performer, at an F-score of 0.865. The differential between the system combination method when trained over auto-generated POS and chunk data (Combine*) and that trained over gold standard data ($Combine^*_{Penn}$) is still tangible, but considerably less than for any of the individual methods. Importantly, Combine* outperforms the gold standard results for each of the individual methods. Examples of false positives (i.e. verb–prepositions misclassified as VPCs) returned by this final system configuration are *firm away*, *base on* and *very off*.

In Section 1, we made the claim that VPCs are highly productive and domain-specific. We validate this claim by comparing the 1000 VPCs correctly extracted by the Combine* method against both the LinGO-ERG and the relatively broad-coverage Alvey Tools VPC inventory. The 28 March, 2002 version of the LinGO-ERG contains a total of 300 intransitive and transitive VPC types, of which 195 were contained in the 1000 correctly-extracted VPCs. Feeding the remaining 805 VPCs into the grammar (with a lexical type describing their transitivity) would therefore result in an almost four-fold increase in the total number of VPCs, and increase the chances of the grammar being able to parse WSJ-style text. The Alvey Tools data contains a total of 2254 VPC types. Of the 1000 extracted VPCs, 284 or slightly over 28%, were not contained in the Alvey data, with examples including *head down*, *blend together* and *bid up*. Combining this result with that for the LinGO-ERG, one can see that we are not simply extracting information already at our fingertips, but are accessing significant numbers of novel VPC types.

[7] Note that only a limited number of VPCs can be deverbalised in this manner: of the 62 VPCs attested in the WSJ, only 8 had a deverbal usage.

7 Related research

There is a moderate amount of research related to the extraction of VPCs, or more generally phrasal verbs, which we briefly describe here.

One of the earliest attempts at extracting "interrupted collocations" (i.e. non-contiguous collocations, including VPCs), was that of Smadja (1993). Smadja based his method on bigrams, but unlike conventional collocation work, described bigrams by way of the triple of $\langle word_1, word_2, posn \rangle$, where *posn* is the number of words occurring between $word_1$ and $word_2$ (up to 4). For VPCs, we can reasonably expect from 0 to 4 words to occur between the verb and the particle, leading to 5 distinct variants of the same VPC and no motivated way of selecting between them. Smadja did not attempt to evaluate his method other than anecdotally, making any comparison with our research impossible.

The work of Blaheta and Johnson (2001) is closer in its objectives to our research, in that it takes a parsed corpus and extracts out multiword verbs (i.e. VPCs and prepositional verbs) through the use of log-linear models. Once again, direct comparison with our results is difficult, as Blaheta and Johnson output a ranked list of all verb–preposition pairs, and subjectively evaluate the quality of different sections of the list. Additionally, they make no attempt to distinguish VPCs from prepositional verbs.

The method which is perhaps closest to ours is that of Kaalep and Muischnek (2002) in extracting Estonian multiword verbs (which are similar to English VPCs in that the components of the multiword verb can be separated by other words). Kaalep and Muischnek apply the "mutual expectation" test over a range of "positioned bigrams", similar to those used by Smadja. They test their method over three different corpora, with results ranging from a precision of 0.21 and recall of 0.86 (F-score=0.34) for the smallest corpus, to a precision of 0.03 and recall of 0.85 (F-score=0.06) for the largest corpus. That is, high levels of noise are evident in the system output, and the F-score values are well below those achieved by our method for English VPCs.

8 Conclusion

In conclusion, this paper has been concerned with the extraction of English verb–particle constructions from raw text corpora. Three basic methods were proposed, based on tagger output, chunker output and a chunk grammar; the chunk grammar method was optionally combined with attachment resolution to determine the syntactic structure of verb–preposition pairs in ambiguous constructs. We then experimented with combining the output of the three methods together into a single classifier, and further complemented the feature space with a number of lexical and frequentistic features, culminating in an F-score of 0.865 over the WSJ.

It is relatively simple to adapt the methods described here to output subcategorisation

types, rather than a binary judgement on verb–particlehood. This would allow the extracted output to be fed directly into the LinGO-ERG for use in parsing. We are also interested in extending the method to extract prepositional verbs, many of which appear in the attachment resolution data and are subsequently filtered out by the consolidated classifier.

Acknowledgements

This research was supported in part by NSF grant BCS-0094638 and also the Research Collaboration between NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation and CSLI, Stanford University. We would like to thank Francis Bond, Ann Copestake, Dan Flickinger, Diana McCarthy and the three anonymous reviewers for their valuable input on this research.

References

Don Blaheta and Mark Johnson. 2001. Unsupervised learning of multi-word verbs. In *Proc. of the ACL/EACL 2001 Workshop on the Computational Extraction, Analysis and Exploitation of Collocations*, pages 54–60.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21:543–65.

Nicoletta Calzolari, Charles J. Fillmore, Ralph Grishman, Nancy Ide, Alessandro Lenci, Catherine MacLeod, and Antonio Zampolli. 2002. Towards best practice for multiword expressions in computational lexicons. In *Proc. of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1934–40.

Ann Copestake and Dan Flickinger. 2000. Open source grammar development environment and broad-coverage English grammar using HPSG. In *Proc. of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, pages 591–8.

Ann Copestake, Fabre Lambeau, Aline Villavicencio, Francis Bond, Timothy Baldwin, Ivan Sag, and Dan Flickinger. 2002. Multiword expressions: linguistic precision and reusability. In *Proc. of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1941–7.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2001. TiMBL: Tilburg Memory Based Learner, version 4.1, reference guide. ILK technical report 01-04.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

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

Stefan T. Gries. 2000. *Towards multifactorial analyses of syntactic variation: The case of particle placement*. Ph.D. thesis, University of Hamburg.

Claire Grover, John Carroll, and Edward Briscoe. 1993. The Alvey Natural Language Tools grammar (4th release). Technical Report 284, Computer Laboratory, Cambridge University, UK.

Rodney Huddleston and Geoffrey K. Pullum. 2002. *The Cambridge Grammar of the English Language*. Cambridge: Cambridge University Press.

Ray Jackendoff. 1997. *The Architecture of the Language Faculty*. Cambridge, MA: MIT Press.

Heiki-Jaan Kaalep and Kadri Muischnek. 2002. Using the text corpus to create a comprehensive list of phrasal verbs. In *Proc. of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, pages 101–5.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–30.

Kathleen R. McKeown and Dragomir R. Radev. 2000. Collocations. In Robert Dale, Hermann Moisl, and Harold Somers, editors, *Handbook of Natural Language Processing*, chapter 21. Marcel Dekker.

Guido Minnen, John Carroll, and Darren Pearce. 2000. Robust, applied morphological generation. In *Proc. of the First International Natural Language Generation Conference (INLG)*, pages 201–8.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. 7(3).

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

Frank Smadja. 1993. Retrieving collocations from text: Xtract. *Computational Linguistics*, 19(1):143–78.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. of the 4th Conference on Computational Natural Language Learning (CoNLL-2000)*, pages 127–132.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*.

Jorn Veenstra and Antal van den Bosch. 2000. Single-classifier memory-based phrase chunking. In *Proc. of the 4th Conference on Computational Natural Language Learning (CoNLL-2000)*, pages 157–9.

Aline Villavicencio and Ann Copestake. 2002a. Phrasal verbs and the LinGO-ERG. *LinGO Working Paper No. 2002-01*.

Aline Villavicencio and Ann Copestake. 2002b. Verb-particle constructions in a computational grammar. In *Proc. of the 9th International Conference on Head-Driven Phrase Structure Grammar (HPSG-2002)*.