Classification of Verb-Particle Constructions with the Google Web1T Corpus

Jonathan K. Kummerfeld and **James R. Curran**
School of Information Technologies
University of Sydney
NSW 2006, Australia
{jkum0593,james}@it.usyd.edu.au

Abstract

Manually maintaining comprehensive databases of multi-word expressions, for example Verb-Particle Constructions (VPCs), is infeasible. We describe a new type level classifier for potential VPCs, which uses information in the Google Web1T corpus to perform a simple linguistic constituency test. Specifically, we consider the fronting test, comparing the frequencies of the two possible orderings of the given verb and particle. Using only a small set of queries for each verb-particle pair, the system was able to achieve an F-score of 75.7% in our evaluation while processing thousands of queries a second.

1 Introduction

Creating comprehensive linguistic resources manually is an expensive and slow process, making it infeasible for maintaining up-to-date resources of constantly evolving linguistic features, such as Multi-Word Expressions (MWEs). These resources are crucial for a range of Natural Language Processing (NLP) tasks, such as accurate parsing. Identification of MWEs does not prevent the production of syntactically accurate parses, but the extra information can improve results. Since manually creating these resources is a challenge, systems are needed that can automatically classify expressions accurately. Also, because these systems will be running during parsing, they need to be fast to prevent the creation of an additional bottleneck.

Here we focus on Verb-Particle Constructions (VPCs), a type of MWE composed of a verb and a particle. Villavicencio (2003a) showed that VPCs are poorly covered by corpus data and constantly growing in number, making them a suitable candidate for an automatic classification system. Previous work on VPCs has mainly focused either on their compositionality (McCarthy et al., 2003), or on using sophisticated parsers to perform extraction from corpora (Baldwin and Villavicencio, 2002). While parser based methods have been very successful (Kim and Baldwin, 2006) they rely on contextual knowledge and complex processing. The Web has been used as a corpus previously by Villavicencio (2003b), who used search engines as a source of statistics for classification, but her aim was to create new resources, rather than a tool that can be used for identification. We have constructed a high-throughput query system for the Google Web1T data, which we use to collect information to perform the fronting linguistic constituency test. The results of this process are used to train a classifier, which can then quickly and accurately classify a given verb-particle pair.

The Google Web1T corpus is over twenty-four gigabytes of plain text in compressed form, making it infeasible to store the entire data-set in memory and perform queries directly. To handle the data we have aggregated the frequency counts for n-grams by using templates that contain a mixture of wild-cards and words, where the words are all possibilities taken from three sets. The new data this process produces is stored in a hash-table distributed across several computers, making fast queries possible. These queries provide the frequency of templates such as `verb ? particle` and `particle ? verb`, which can then be compared as a form of the fronting test. By comparing the frequency of these templates we are able to construct classifiers for verb-particle pairs with an accuracy of 74.7%, recall of 74.7%, and a precision of 76.8%. These results indicate that this method is a promising source of information for fast on-line classification of verb-particle pairs as VPCs.

2 Background

VPCs are MWEs composed of a verb and a particle, where particles are derived from two categories, prepositions and spatial adverbs (Quirk et al., 1985). However, most research on VPCs has focused on prepositions only, because they are more productive. Semantically, the compositionality of multi-word expressions varies greatly. Often they have the property of paraphrasability, being replacable by a single verb of equivalent meaning, and exhibit prosody, having a distinctive stress pattern.

Previous work concerning VPCs can be broadly divided into two groups, compositionality analysis, and classification. Our work is entirely concerned with classification, but we will briefly describe previous work on compositionality as the two areas are closely linked.

2.1 Compositionality

Determining how much the simplex meaning of individual words in a MWE contribute to the overall meaning is challenging, and important for producing semantically correct analysis of text. A range of methods have been considered to differentiate examples based on their degree of semantic idiosyncrasy.

Initial work by Lin (1999) considered the compositionality of MWEs by comparing the distributional characteristics for a given MWE and potentially similar expressions formed by synonym substitution. This was followed by Bannard et al. (2003), who used human non-experts to construct a gold standard dataset of VPCs and their compositionality, which was used to construct a classifier that could judge whether the two words used contributed their simplex meaning. McCarthy et al. (2003) used an automatically acquired thesaurus in the calculation of statistics regarding the compositionality of VPCs and compared their results with statistics commonly used for extracting multiwords, such as latent semantic analysis. Bannard (2005) compared the lexical contexts of VPCs and their component words across a corpus to determine which words are contributing an independent meaning.

Light Verb Constructions (LVCs) are another example of an MWE that has been considered in compositionality studies. The challenge of distinguishing LVCs from idioms was considered by Fazly et al. (2005), who proposed a set of statistical measures to quantify properties that relate to the compositionality of an MWE, ideas that were then extended in Fazly and Stevenson (2007).

Recently, Cook and Stevenson (2006) addressed the question of which sense of the component words is being used in a particular VPC. They focused on the contribution by particles and constructed a feature set based on the properties of VPCs and compared their effectiveness with standard co-occurrence measurements.

2.2 Classification

A range of methods for automatic classification of MWEs have been studied previously, in particular the use of parsers, applying heuristics to n-grams, and search engine queries.

Possibly the earliest attempt at classification, was by Smadja (1993), who considered verb-particle pairs, separated by up to four other words, but did not perform a rigorous evaluation, which prevents us from performing a comparison.

Recent work has focused on using parsers over raw text to gain extra information about the context of the verb-particle pair being considered. In particular, the information needed to identify the head noun phrase (NP) for each potential VPC. Early work by Blaheta and Johnson (2001) used a parsed corpus and log-linear models to identify VPCs. This was followed by Baldwin and Villavicencio (2002) who used a range of parser outputs and other features to produce a better informed classifier. Other forms of linguistic and statistical unsupervised methods were considered by Baldwin (2005), such as Pointwise Mutual Information. This work was further extended by Kim and Baldwin (2006) to utilise the sentential context of verb-particle pairs and their associated NP to improve results.

The closest work to our own in terms of the corpus used is that of Villavicencio (2003b), in which the web is used to test the validity of candidate VPCs. Also, like our work, Villavicencio does not consider the context of the verb and particle, unlike the parser based methods described above. A collection of verb-particle pairs was generated by combining verbs from Levin's classes (Levin,

1993) with the particle up. Each potential VPC was passed to the search engine Google to obtain an approximate measure of their frequency, first on their own, then in the form `Verb up for` to prevent the inclusion of prepositional verb forms. The measurement is only approximate because the value is a document count, not an actual frequency as in the Web1T data. This led to the identification of many new VPCs, but the results were not evaluated beyond measurements of the number of identified VPCs attested in current resources, making comparison with our own work difficult.

When a verb-particle pair is being classified the possibilities other than a VPC are a prepositional verb, or a free verb-preposition combination. A variety of tests exist for determining which of these a candidate verb-particle pair is. For the transitive case, Baldwin and Villavicencio (2002) applied three tests. First, that VPCs are able to undergo particle alternation, providing the example that hand in the paper and hand the paper in are both valid, while refer to the book is, but *refer the book to is not. Second, that pronominal objects must be expressed in the split configuration, providing the example that hand it in is valid, where as *hand in it is not. And finally, that manner adverbs cannot occur between the verb and particle. The first two of these tests are context based, and therefore not directly usable in our work, and the third is particularly specific. Instead, for now we have focused on the 'fronting' constituency test, which involves a simple rearrangement of the phrase, for example hand in the paper would be compared to *in the paper hand.

2.3 The Google Web1T corpus

The Google Web1T corpus is an extremely large collection of n-grams, extracted from slightly more than one trillion words of English from public web pages (Brants and Franz, 2006). This dataset poses new challenges for data processing systems, as it is more than twenty-four gigabytes in compressed form.

The corpus contains n-grams up to 5-grams, such as in Table 1. This example considers the pair of words 'ferret' and 'out', showing all entries in the corpus that start and end with the two words, excluding those that contain non-alphabetical characters.

N-gram	Frequency
ferret out	79728
out ferret	74
ferret her out	52
ferret him out	342
ferret information out	43
ferret is out	54
ferret it out	1562
ferret me out	58
ferret that out	180
ferret them out	1582
ferret these out	232
ferret things out	58
ferret this out	148
ferret you out	100
out a ferret	63
out of ferret	71
out the ferret	120
ferret it all out	47
ferret these people out	60
ferret these projects out	52
out of the ferret	54
ferret lovers can ferret out	45
out a needing shelter ferret	63

Table 1: N-grams in the Web1T corpus for the VPC ferret out.

One approach to this dataset is to simply treat it as a normal collection of n-gram frequencies, scanning the relevant sections for answers, such as in Bergsma et al. (2008) and Yuret (2007). Another approach is used by Talbot and Brants (2008), pruning the corpus to a third of its size and quantising the rest, reducing the space used by frequency counts to eight bits each. These methods have the disadvantages of slow execution and only providing approximate frequencies respectively.

Hawker et al. (2007) considered two methods for making practical queries possible, pre-processing of the data, and pre-processing of queries. The first approach is to reduce the size of the dataset by decreasing the resolution of measurements, and by accessing n-grams by implicit information based on their location in the compressed data, rather than their actual representa-

tion. While this avoids the cost of processing the entire dataset for each query, it does produce less accurate results. The second method described is intelligent batching queries, then performing a single pass through the data to answer them all.

The Web1T corpus is not the only resource for n-gram counts on the web. Lapata and Keller (2005) use Web counts, the number of hits for a search term, as a measure of the frequency of n-grams. This approach has the disadvantage that the frequency returned is actually a document frequency, where as the Web1T values are counts of the occurrences of the actual n-gram, including repetitions in a single page.

2.4 Weka

Weka is a collection of machine learning algorithms for data mining tasks (Witten and Frank, 2005). From the range of tools available we have used a selection of the classifiers: ZeroR, OneR, Id3, J48, Naive Bayes and Multilayer Perceptron.

As a baseline measure we used the ZeroR classifier, which always returns the mode of the set of nominal options. Next we considered the OneR classifier, which compares the information gain of each of the features and chooses the most effective. Two decision tree classifiers were considered, Id3 (Quinlan, 1993b) and J48 (Quinlan, 1993a). Both initially construct the tree in the same way, using information gain to choose the feature to split on next, until either no features remain, or all samples are of the same class. The difference is that J48 attempts to prune the tree.

We also considered two non-decision based classifiers, Naive Bayes and Multilayer Perceptron. The Naive Bayes classifier assumes features are independent and applies Bayes' Theorem to calculate the probability of an example belonging to a particular class. The Multilayer Perceptron uses multiple layers of sigmoid nodes, whose links are adjusted during training by back-propagation of corrections (Rumelhart et al., 1986).

3 Classifying verb-particle pairs

Our aim was to construct a system capable of classifying a given verb-particle pair as a potential VPC or not. To do this we currently apply the fronting test to the verb-particle pair. We describe the classification given by the system as 'potential' because the system only uses the verb-particle pair and not the specific context being considered. The system also needed a simple interface for making fast queries.

The fronting test is a linguistic constituency test that gauges whether words function as a single unit in a phrase by breaking the phrase in half between the words and swapping the order of the two halves (Quirk et al., 1985). In the case of VPCs this means the following rearrangement (note that $\texttt{NP}_2$ may not always be present):

- $\texttt{NP}_1$ Verb Particle $\texttt{NP}_2$
- Particle $\texttt{NP}_2$ $\texttt{NP}_1$ Verb

If the rearranged form of a phrase is also valid it implies the verb and particle are not functioning as a single unit in the phrase, and so we do not classify them as a VPC.

In this structure there are two forms of verbs to consider. For the intransitive case a VPC always results (Baldwin and Villavicencio, 2002), for example:

- He gets around.
- *Around he gets.

However, in the transitive case there are three possibilities, for example:

- The mafia ran down Joe.
- *Down Joe the mafia ran.
- What did you refer to?
- To what did you refer?
- I walked to the house.
- To the house, I walked.

The last two examples are not VPCs, but rather examples of a prepositional verb and a free verb-preposition combination respectively. Note that both orders are valid for these, where as for the VPCs the second is not.

We use the frequency counts in the Web1T data to measure the validity of particular orderings. For this purpose the meaning of the absolute value returned is difficult to quantify, so instead we compare the frequencies of multiple arrangements of the same verb-particle pair. The more common arrangement is treated as being more valid. The exact form of these comparisons is described in the following sections.

4 Implementation

4.1 Structure

At the core of our system is a hash table that allows random access to the parts of the Web1T data that are relevant to our queries. The table uses linear probing based on a 64bit key that is a numerical representation of each word pattern, interpreted in base thirty-seven. The initial position considered is the key value modulo the size of our table (which may vary to allow support for different amounts of system RAM). While there is the possibility of collisions, the probability of their occurrence is extremely small, and none were observed during testing.

Word patterns are used to generate the key, and at the location specified by the key (or a position that is probed to) we store the complete key and four integers. The system is designed to process any string to generate the key, rather than specifically VPCs, and the four integers stored are returned directly, to be manipulated as desired. We use these four integers to hold aggregated counts and the string used is based on word patterns containing wildcards.

The word patterns used to create the keys may contain two types of symbols, a wildcard, or a set reference. Wildcards represent any word not in the given sets, where we do not include punctuation, numbers or sentence boundaries as words. Set references indicate that the word at that position must come from one of the provided sets. Currently the system handles up to three sets, each of which can be referenced at most once. Also note that patterns with leading or trailing wildcards do not need to be considered, since they are already included in the count for the equivalent n-gram without the leading or trailing wildcards.

For two sets, the four patterns used are as follows, where `_` represents a wildcard, and `A` and `B` are set references:

1. `A B`
2. `A _ B`
3. `A _ _ B`
4. `A _ _ _ B`

For three sets we cannot store the frequency of each pattern separately as there are six possible patterns, but only four spaces in each entry of the table. We chose to aggregate the values further, accepting the following patterns:

1. `A B C`
2. `A B _ C` or `A B _ _ C`
3. `A _ B C` or `A _ _ B C`
4. `A _ B _ C`

All data in the Web1T corpus that does not conform to one of these patterns is ignored. This means the size of the hash table needed, and therefore the amount of memory needed, depends entirely on the sizes of the word sets provided.

The hash table, word sets, and their interface, are written in C++, but are wrapped up inside a Python extension. This python extension is used by a server-client interface, where each server holds a single hash table that has all the aggregated counts for words within a specific alphabetic range. To perform queries a client connects to a set of servers, receives their ranges, and then requests frequency counts directly from the appropriate server. This structure effectively increases the size of the system memory that is accessible, but also has the side effect that multiple clients can query the system at the same time at high speed.

4.2 Statistics

We performed our experiments using three sets of words, verbs, particles and pronouns. The sizes and sources of the sets were:

- Verbs - 19,046 - all words with a VB based tag in the Wall Street Journal section of the Penn Treebank
- Possible Particles - 257 - all words with either an IN or RP tag in the Wall Street Journal section of the Penn Treebank
- Pronouns - 77 - from Wikipedia

This leads to 2,261,407,764 possible permutations of the words in the word sets. The storage cost for each permutation is 24 bytes, eight of which is for the 64bit key, and the other sixteen bytes are for the four integers, which occupy four bytes each. However, because most of these permutations are not found in the Web1T data, a total of only approximately 2.8 Gigabytes of system memory was required.

Pattern	Frequency
Verb Particle	2 436 566
Verb _ Particle	747 492
Verb _ _ Particle	78 569
Verb _ _ _ Particle	19 549
Particle Verb	2 606 582
Particle _ Verb	2 326 720
Particle _ _ Verb	68 540
Particle _ _ _ Verb	14 016

Table 2: Word patterns and example aggregated frequencies for the VPC hand in.

Since one of our aims was to produce a quick method for classification, the speed of the system is also worth considering. Once running, the servers were able to handle between five and ten thousand queries per second. The time to start the servers, which includes reading all of the Web1T data, was between sixty and ninety minutes. However, this time could be drastically reduced for subsequent runs by writing the constructed hash table out to disk and reading it directly into memory the next time the system needs to be started.

4.3 Hash table queries for VPCs

Once the hash table is constructed the system is able to take two or three words and return the four values corresponding to their position in the hash table (where the words are considered in the order they are given). A simplified form of the fronting test considers the patterns in Table 2.

The word patterns in the top half of Table 2 correspond to $\texttt{NP}_1$ Verb Particle $\texttt{NP}_2$. We have considered the cases with wildcards between the verb and particle since most VPCs can occur in the split configuration. The patterns in the bottom half of Table 2 correspond to Particle $\texttt{NP}_2$ $\texttt{NP}_1$ Verb. We allow the case with a single wildcard to cover examples without a second NP. To get the vales for these patterns the system is queried twice, once with Verb Particle and then with Particle Verb.

As mentioned previously, it is difficult to quantify the absolute values returned by these queries, but by comparing them we are able to measure the relative validity of the two orderings, as described in the following section.

Name	Comparison
vp-1	v ? ? ? p > p _ _ ? v
vp-2	v p > p _ v
nv_p-1	n v ? ? p > p n v
nv_p-2	n v ? ? p > p ? ? n v
n_vp-1	n _ ? v ? p > p n _ ? v
n_vp-2	n _ ? v ? p > p ? n _ ? v

Table 3: Comparisons for judging whether a verb and particle form a VPC.

One problem with this comparison is that the contributions to the values could be coming from different uses of the pair, such as in Table 1, where ferret is out and ferret it out will both contribute to the frequency for the pattern Verb _ Particle. To deal with this we added queries that used the three set patterns. As the third set we used pronouns, which can approximate single word NPs. This method does have the drawback that frequencies will be lower, leading to more unattested examples, but does provide extra information in most cases.

5 Results

For the purposes of classification we created a set of questions that have true or false answers and can be answered based on queries to our system. In many of these cases we consider the sum of a collection of values from our hash table. To keep the descriptions here concise we have used the symbol '?' to denote the optional presence of another wildcard. The value for patterns containing '?'s is the sum of all the possible patterns, with or without wildcards at those positions. Also, rather than using the entire words, we have used 'v' for Verbs, 'p' for Particles and 'n' for pronouns.

To produce a set of invalid VPCs we used our lists of verbs and particles to generate a random list of two thousand verb-particle pairs not in our set of VPCs. The set of two thousand true VPCs was constructed randomly from a large set containing 116 from McCarthy et al. (2003) and 7140 from Baldwin and Villavicencio (2002).

For every pair a feature vector containing the results of these comparisons was created, as demonstrated in Table 4, containing the answers to the questions described previously. Weka was

Test	Comparison	Result
vp-1	3 282 176 > 82 556	True
vp-2	2 436 566 > 2 326 720	True
nv_p-1	435 822 > 1 819 931	False
nv_p-2	435 822 > 1 819 931	False
n_vp-1	0 > 0	False
n_vp-2	0 > 0	False

Table 4: Example comparisons for the VPC hand in.

then used to generate and evaluate a range of classification techniques. All methods were applied with the standard settings and using ten fold cross-validation.

6 Discussion

This lightweight method of classification does not match up to results from parser based methods, such as Kim and Baldwin (2006), who acheived an F-score of 97.4%. However, this high performance relies on context awareness and only applied to the identification of frequent VPC examples. In contrast, by drawing on the Web1T dataset our system overcomes the data sparseness problem. The cost of attempting to classify without knowledge of context is that we apply a linguistic poor approach, achieving lower performance, but there is plenty of scope for extension. The system as it stands is very flexible and can easily scale to support larger sets of words, making a broader range of queries possible. Also, the post-processing of query results could become more sophisticated, and the classifiers used could be tweaked to be more effective. The system itself could also be modified to store other information, making different types of queries possible.

One challenge that remains unaddressed is how to handle multiple uses of the same word pair. For example, the following are both valid:

- The deadline is coming up.
- Joe is an up and coming athlete.

In these two cases the words coming and up have different semantic meanings, but both forms will contribute to the frequency counts for the word pair. While this does skew the results of our queries, we should be able to limit the effect by using the third set to constrain how the words are being used.

Despite the support for three sets of words, we only really have flexibility in one of them, as we must use the other two for Verbs and Particles. Here we chose Pronouns because they could act as single word NPs, allowing us to perform the fronting test more effectively. However, there are other linguistic tests for VPCs. In particular, two of the tests performed by Baldwin and Villavicencio (2002) could be adapted to our system. One is that transitive VPCs must be expressed in the split configuration when the second NP is pronominal, such as hand it in, which is valid, and *hand in it, which is not. Our third set already contains pronouns, all that needs to be done is a comparison of the frequency of `Verb Pronoun Particle` and `Verb Particle Pronoun`, giving a measure of which is more valid. The other test is that manner adverbs cannot occur between the verb and the particle. This could be tested by expanding our third set to include manner adverbs, and then comparing the frequency of `Verb Manner Adverb Particle` and `Verb Particle`.

So far we have only considered a small number of ways to combine the results of the queries to produce features for classifiers. All of the features we considered here were true/false answers to simple questions, constructed intuitively, but without much experimentation. By breaking up our training set and using a part of it to explore the statistical properties of the results from queries we may be able to identify unexpected, but effective, discriminating factors. We may also be able to improve performance by considering a broader range of classifiers, and by adjusting their input parameters to suit our task.

Finally, the system itself could be modified to store more information and to support other types of queries. The corpus does contain sentence boundary markers, and we could use capitalisation to extend this information approximately without wasting a word in the n-gram. Currently these are entirely ignored, except when a sentence boundary occurs in the middle of an n-gram, in which case the n-gram is excluded from the frequency count. Also, because the current system uses linear probing, the entire hash table is stored

Classifier	Accuracy	Precision	Recall	F score
ZeroR	50.0%	25.0%	50.0%	33.3%
OneR	74.4%	76.4%	74.4%	75.4%
Id3	74.7%	76.8%	74.7%	75.7%
J48	74.4%	76.4%	74.4%	75.4%
Naive Bayes	74.7%	76.8%	74.7%	75.7%
MultilayerPerceptron	73.4%	73.6%	73.4%	73.5%

Table 5: Results using Weka on simple query answers.

as a single block of memory. This means the fread and fwrite functions could easily be used to save it on disk, which would drastically reduce the system restart time (assuming the same word sets are being used).

7 Conclusion

We have constructed a system for quickly querying the Web1T corpus, providing the information needed to perform the fronting linguistic constituency test. By performing this test on potential VPCs, on their own, and in combination with a range of pronouns, we produce a feature vector that can be used to construct a classifier. The classifiers produced are accurate and fast, as the only information they need can be quickly accessed from the Web1T corpus by our system. The variation in performance between classifiers is small, but the best performance is by the Id3 and Naive Bayes classifiers, which have a recall of 74.7%, and precision of 76.8%.

A range of other linguistic tests exist, which our system could be extended to support. Adding this extra information to our feature vectors and exploring classification methods further could lead to improved performance. Another area for further investigation is the interpretation of the raw frequency counts and the way comparisons are performed.

Our system is capable of scaling to support a larger proportion of the Web1T data, while retaining the ability to respond quickly to many queries. Also, by adjusting the word sets provided and/or the templates that are used to aggregate frequency counts, our system could be altered to support different types of queries. This flexibility means it could be useful in other tasks that involve querying the Web1T corpus.

8 Acknowledgement

We would like to thank the anonymous reviewers for their helpful feedback and corrections. J. K. K. thanks The University of Sydney for the support of a Merit Scholarship and the Talented Student Program that enabled this work.

References

Timothy Baldwin and Aline Villavicencio. 2002. Extracting the unextractable: a case study on verb-particles. In *Proceedings of the 2002 Conference on Natural Language Learning*, pages 1–7, Taipei, Taiwan, August.

Timothy Baldwin. 2005. Looking for prepositional verbs in corpus data. In *Proceedings of the 2005 Meeting of the Asociation for Computational Linguistics: Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications*, pages 115–126, Colchester, UK, April.

Colin Bannard, Timothy Baldwin, and Alex Lascarides. 2003. A statistical approach to the semantics of verb-particles. In *Proceedings of the 2003 Meeting of the Asociation for Computational Linguistics: Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 65–72, Sapporo, Japan, July.

Colin Bannard. 2005. Learning about the meaning of verb-particle constructions from corpora. *Journal of Computer Speech and Language*, 19(4):467–478.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Distributional identification of non-referential pronouns. In *Proceedings of the 2008 Meeting of the Asociation for Computational Linguistics: Human Languages Technologies*, pages 10–18, Columbus, Ohio, June.

Don Blaheta and Mark Johnson. 2001. Unsupervised learning of multi-word verbs. In *Proceedings of the 2001 Meeting of the Asociation for Computational Linguistics: Workshop on Collocation the Compu-*

tational Extraction, Analysis and Exploitation of Collocations, pages 54–60, Toulouse, France, July.

Thorsten Brants and Alex Franz. 2006. Web1t 5-gram corpus version 1.1. Technical report, Google Research.

Paul Cook and Suzanne Stevenson. 2006. Classifying particle semantics in English verb-particle constructions. In *Proceedings of the 2006 Meeting of the Association for Computational Linguistics and the International Committee on Computational Linguistics Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 45–53, Sydney, Australia, July.

Afsaneh Fazly and Suzanne Stevenson. 2007. Distinguishing subtypes of multiword expressions using linguistically-motivated statistical neasures. In *Proceedings of the 2007 Meeting of the Asociation for Computational Linguistics: Workshop on A Broader Perspective on Multiword Expressions*, pages 9–16, Prague, Czech Republic, June.

Afsaneh Fazly, Ryan North, and Suzanne Stevenson. 2005. Automatically distinguishing literal and figurative usages of highly polysemous verbs. In *Proceedings of the 2005 Meeting of the Asociation for Computational Linguistics: Workshop on Deep Lexical Acquisition*, Ann Arbor, USA, July.

Tobias Hawker, Mary Gardiner, and Andrew Bennetts. 2007. Practical queries of a massive n-gram database. In *Proceedings of the 2007 Australasian Language Technology Workshop 2007*, pages 40–48, Melbourne, Australia, December.

Su Nam Kim and Timothy Baldwin. 2006. Automatic identification of english verb particle constructions using linguistic features. In *Proceedings of the 2006 Meeting of the Asociation for Computational Linguistics: Workshop on Prepositions*, pages 65–72, Trento, Italy, April.

Mirella Lapata and Frank Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1):1–33.

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press.

Dekang Lin. 1999. Automatic identification of non-compositional phrases. In *Proceedings of the 1999 Annual Meeting of the Association for Computational Linguistics*, pages 317–324, College Park, Maryland, USA.

Diana McCarthy, Bill Keller, , and John Carroll. 2003. Detecting a continuum of compositionality in phrasal verbs. In *Proceedings of the 2003 Meeting of the Asociation for Computational Linguistics: Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, Sapporo, Japan, July.

John Ross Quinlan. 1993a. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

John Ross Quinlan, 1993b. *Induction of decision trees*, pages 349–361. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Randolph Quirk, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, London.

David Rumelhart, Geoffrey Hinton, and Ronald Williams, 1986. *Learning internal representations by error propagation*, pages 318–362. MIT Press, Cambridge, MA, USA.

Frank Smadja. 1993. Retrieving collocations from text: Xtract. *Journal of Computational Linguistics*, 19(1):143–177.

David Talbot and Thorsten Brants. 2008. Randomized language models via perfect hash functions. In *Proceedings of the 2008 Meeting of the Asociation for Computational Linguistics: Human Languages Technologies*, pages 505–513, Columbus, Ohio, June.

Aline Villavicencio. 2003a. Verb-particle constructions and lexical resources. In *Proceedings of the Meeting of the Asociation for Computational Linguistics: 2003 workshop on Multiword expressions*, pages 57–64, Sapporo, Japan, July.

Aline Villavicencio. 2003b. Verb-particle constructions in the world wide web. In *Proceedings of the 2003 Meeting of the Asociation for Computational Linguistics: Workshop on the Linguistic Dimensions of Prepositions and their use in Computational Linguistics Formalisms and Applications*, Sapporo, Japan, July.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Fransisco, 2nd edition.

Deniz Yuret. 2007. KU: Word sense disambiguation by substitution. In *Proceedings of the 2007 International Workshop on Semantic Evaluations*, Prague, Czech Republic, June.