Classifying Particle Semantics in English Verb-Particle Constructions

Samin Fakharian
samin.fakharian@unb.ca

Abstract

Nowadays Multiword expressions (MWEs) include a wide range of phenomena such as fixed expressions, idioms, and light verb constructions. MWEs are a key problem for the development of a large scale, linguistically sound natural language processing technology. As verb-particle constructions (VPCs) are a rich source of MWEs in English, many works focused on their compositionality. We decide to address the issue of finding which meaning of a component word is being used in the VPCs by classifying the particle semantics, in this work the particle ”up”. We develop a feature space for use in our classification of the sense the particle contributed in a VPC. For our feature space we re-implement the particle features from our baseline paper and also we used multiple word embedding methods. Our results show that our features mostly outperform the slot features used by the base paper in our classification task on unseen test VPCs.

1 Introduction

Nowadays Multiword expressions (MWEs) include a wide range of phenomena such as fixed expressions, idioms, and light verb constructions. MWEs are a key problem for the development of a large scale, linguistically sound natural language processing technology. By focusing on the compositionality of verb-particle constructions (VPC) we decide to work on classifying the meaning of the component words that are being used in each VPC with the particle ”up” by using word embeddings and the particle features. Multiword Expressions (MWEs) are expressions that can be syntactically and/or semantically idiosyncratic and are composed of at least 2 words. In fact, in a text, they act as a single unit. As MWEs are commonly used in language they are important for the natural language processing community and their problem that they pose is still an open research area. MWEs are a challenge for both natural language applications and linguistic theory (Copestake et al., 2002) as we have a challenge in learning the semantics of multiword expressions as they vary in the degree of compositionality (Salton et al., 2016). We have to decide how much each component of multiword expression contributes to the overall meaning semantics of the expression.

Recent NLP work on semantic idiomaticity focuses on predicting the compositionality of MWEs whereby a given MWE is mapped onto a continuous-values compositionality score (Reddy et al., 2011). However, the works in this area do not answer to the question that which meaning of a component word is being used when MWE is compositional. As a word is mostly ambiguous, so that even if we find that a multiword expression is compositional, the meaning of the MWE is still unknown as we have to find out the actual meaning the components convey.

In another area of the research, the NLP community now is focusing on the learning distributed representations of word meaning in the form of word embeddings. Pre-trained word embeddings have probed to be significantly useful in most of downstream NLP tasks such as sentiment analysis (Tang et al., 2015), named-entity recognition (Collobert and Weston, 2008), machine translation (Zou et al., 2013), and so on. Although the embedding models have been shown to work significantly well across a range of tasks (Peters et al., 2018) the study of their performance on data with varying degrees of compositionality whose meaning is also not easily predicted from that of its constituent words has been limited (Salehi et al., 2015). Word2vec which is proposed by (Mikolov et al., 2013) is among the first word representations which have been widely used. Classification of the meaning of MWEs is an important issue. Even if an MWE is compositional we still can not decide on its meaning. In this paper, we focused on the verb-particle constructions.

Our first primary question is finding which meaning of a component word actually contributes to a given MWEs. As VPCs with the particle "up" are very frequent and exhibit a wide range of meanings we decided to use them for our experiment. The second question we want to know is whether word embeddings are pre-trained word embeddings are superior to our baseline features space they manually built. Also whether combining word embedding with particle features which we will discuss later in the chapters are going to improve the result or not. Our primary goal is exploring the particle's meaning automatically when it is being used with a given verb in VPC.

We address the issue of automatically determining the semantic contribution of the particle in a verb-particle construction (VPC). We assume that every VPC is compositional and that the semantic contribution of a particle corresponds to one of the small numbers of senses. In this study, we applied various embedding methods to the task of determining which sense of a particle is contributing to the semantic of VPC. As a feature space, we used word embedding feature space to capture semantic and syntactic properties of verbs and VPCs for type classification of English VPCs according to the sense contributed by their particle. Our feature space draws on the general properties of VPCs and is not specific to the particle "up" we decide to work on. Our result shows that by using pre-trained word embeddings as features and using different classifiers we get competitive results with our baseline. A VPC can be ambiguous because its particle can occur in more that one sense. In order to perform type classification, we used multiple classifiers like logistic regression and linear Support Vector Machine (SVM) and we compare our result to the work of (Cook and Stevenson, 2006) that we take as our base model. We can see some improvements over the baseline and combining word embedding features with particle features gives us results on par with the baseline. We show that despite their success on a range of features they manually created, a simple word2vec works better than their slot features.

2 Related Work

A lot of works have been done in the NLP that focused on multiword expressions. There are multiple annotated resources made available in the last few years thanks to (Schneider et al., 2016) that helped the supervised MWE identification. Also, much research on MWE identification has focused on specific kinds of MWEs like (Patrick and Fletcher, 2005).

The paper proposed by (Saied et al., 2019) described and compared the development and tuning of linear versus neural classifiers to use in the systems for MWE identification. They used a simple feed-forward network with one hidden layer.

Since (Collobert and Weston, 2008) proposed a unified neural network architecture to learn distributed word representations and demonstrated its performance on downstream tasks, with word2vec (Mikolov et al., 2013) being the catalyst to the "embedding revolution", embedding learning starts to be utilized in NLP. Language embeddings are an instance of using unsupervised representation learning and they are mostly preferred because they can be learned from unlabeled corpora and we can eliminate the expense of annotation which are also time-consuming.

Applying word embeddings was used by (Salehi et al., 2015) to predict the compositionality of MWEs. They assume that the compositionality of the MWE is related to the relative similarities between each variable and the total MWE expressed by their respective embeddings. Also in another word done by (Salehi et al., 2015) using word embeddings to predict the compositionality of multiword expressions with both single- and multi-prototype word embeddings show that methods using word embeddings outperform count-based distributional similarity.

A work done by (Nandakumar et al., 2019) proposed to apply various embedding methods to multiword expressions to study the degree that multiword expressions can capture the subtle information of non-compositional data following the work of (Salehi et al., 2015) and (Nandakumar et al., 2018). They utilized multiple levels of embeddings and they showed that the best result came from using word2vec. They experiment with all multiple contextualized embedding models like BERT and ELMO and they find out that we can not capture useful information about the non-compositionality of MWEs using these models.

In the work of (McCarthy et al., 2003), the overall compositionality of VPCs was measured automatically and rated. After that(Bannard, 2005)considered the extent to which the verb and particle each contribute semantically to the VPC. Also, some works have been dedicated to sensing disambiguation of specific prepositions (Alam, 2004), and other works like (O'Hara and Wiebe, 2003) has classified preposition tokens according to their semantic tole. The most similar task to ours that is also our baseline is the work of (Cook and Stevenson, 2006) which they developed their feature space based on syntactic and semantic properties of VPCs and classify the particle semantics using SVM with the radial basis function as its kernel type specifically for the VPCs with the "up" as their particles.

3 Proposed Model

In our work, the system aims to classify the particle semantics in English verb-particle constructions. We base our model on the work of (Cook and Stevenson, 2006) that they create their own feature space based on linguistically motivated features and word-co occurrence features. We focused on their linguistical features and we used word embeddings to capture syntactic expressions of the verb also we re-implement their particle features. We continued their work and we combine these features and compare our results with them. Following (Cook and Stevenson, 2006) work we compute linguistic features that are derived by specific semantic and syntactic properties of verbs and VPCs. We also think that the semantic contribution of a particle is related to the semantics of that verb when it comes after the verb. The presence of these trends indicates that features that are effective for the semantic classification of verbs may be useful for our work.

Unlike the work by (Cook and Stevenson, 2006) we made use of various word-level embedding methods to capture features that are related to the syntactic expression of a verb and the participants in the event verb describes. We hypothesize that our word embeddings can work at least on par with the work of the base paper to capture slot features as the word embeddings are capturing the same concepts they used in their model. Where available, we made use of pre-trained models as is standard practice in NLP. As the different models were trained on different corpora, we do not want to compare the different models. We also train our word embedding model on the British National Corpus (BNC).

We re-implement the particle features (Cook and Stevenson, 2006) used in their paper. In (Wurmbrand, 2000), the authors claimed that the compositional particle verbs in German, which are a lot like a phenomenon to English VPCs, accept other particles with similar semantics as their particle. The work by (Cook and Stevenson, 2006) extend this idea, hypothesizing that when a verb combines with a particle such as up in a particular sense, the pattern of usage of that verb in VPCs using all other particles may be indicative of the sense of the target particle (in this case up) when combined with that verb. To reflect this observation, we have to count the relative frequency of any occurrence of the verb used in a verb-particle construction with each of a set of high-frequency particles that were mentioned in their paper. The high-frequency particles were extracted from the British national corpus. Second, as VPCs can often occur in either the joined configuration or the split configuration (Fraser, 1974) notes that when the sense of the particle is not literal it is more possible that it occurs in the joined construction. To test this idea, we calculate the relative frequency of the verb co-occurring with the particle up with each of to reflect that words between the verb and up, reflecting varying degrees of verb-particle separation.

In the next step we classify the features extracted from our dataset with multiple classifiers. In our work we decide to get the result from word embeddings features and particle features. We utilize pre-trained word embeddings and also we get the word embeddings of the British National Corpus (BNC) dataset. We take into consideration multiple way a representing the verbs in our model with BNC. First we take the general form of the words and get the word embeddings of the words. In the next step we take the part of speech tagged of the words in the BNC and tag the verbs and in another step identify the verb part of the verb-particle construction. We also go further and identify the verb-particle constructions with the particle "up" and tagged them properly to compare how the different representations of the words will help us with our results. Then we used multiple classifiers with the feature space we discussed earlier. For our evaluation we decided to use the same metric as (Cook and Stevenson, 2006), the macro-average recall, and we compared our result with them.

4 Material and Methods

We used a list of English VPCs using *up* that (Cook and Stevenson, 2006) created for their experiment. The list they gathered was based on a list of VPCs made available by jackendoff2002english, and a list of VPC compiled by two human judges. Out of 389 VPCs, they gathered they chose for each the training, verification, and test data, 60 VPCs randomly selected from the list. They claimed that as the expenses of manually annotating the data was a lot, they decide to pick only 60 VPCs for each category. The verification data is being utilized in the exploration of the feature space and it helped us to choose the parameters which suit our work better and the final features we choose in the testing stage. For the comparison purposes between our work and the baseline, we pick the same dataset as the baseline paper. We held out the test data for the final testing of the classifiers.

We made use of various word-level embedding methods. Where available we made use of pre-trained versions of each as is standard practice in NLP. We also retrain our model over a standard dataset of the British National Corpus and compare our model results with the pre-trained ones. As we stated before, a word embedding captures the context of a word in a document in the form of vector representation and it tokenizes text at the word level. We trained word2vec (Mikolov et al., 2013) on a British National Corpus. We decided to remove formatting and punctuation and keep the word form of each word. We get the average of the embeddings for each word to get the single embedding for each word. For more analysis purposes we used both the CBOW and SkipGram model and we further compare the results of both of these models for each step. In the CBOW model, the distributed representations of context (or surrounding words) are combined to predict the word in the middle. While in the SkipGram model, the distributed representation of the input word is used to predict the context.

Like the work of (Cook and Stevenson, 2006) we extract our features from the 100M word British National Corpus (BNC, (Burnard, 1995)). In the first step we did not look at the part-of-speech tags of any word and we get the word representations with the word2vec model. We decide to take the 300-dimensional feature vector space for each word occurring more than 5 times in our dataset. In the next step, we decide to utilize the part-of-speech tags and for each word that was verb we added a "_V" symbol at the end to all the words. Then we retrain our word2vec and get a new vector representation for each word. In this way, we can capture the semantics of the base verb of the target VPC, and thus the semantics of the target. After this step, we decide to identify the verb which is considered a component word in VPCs. The use of a verb is considered as a verb-particle construction if it occurs with a particle (tagged AVP) within a six-word window to the right. Now that we find the words that are tagged as a verb and they are part of VPCs, in this step, we add a "_VPC" tag at the end of them and retrain the word2vec model once again. In this way we want to test whether directly learn about the semantics of the target VPC, is going to affect the result or not. Next, we add a symbol "_VPCUP" to the words that are tagged as a verb and they take "up" as their particle to gain information about the semantics of the base verb of the target VPC when used in VPCs with "up".

The particle features are calculated as we discussed earlier. To re-implement the particle features we take 15 particles with the highest frequencies (Cook and Stevenson, 2006). We count how many times the root verb of each target VPC appears in a VPC, occurs with each of the fifteen sets of high-frequency particles, and then we divide the number by the total number of times the base verb of the target VPC occurs. Then we count the number of times the target VPC occurs with each of 0-5 words between the verb and particle, and divide by the total number of times the target VPC occurs. These calculations give us 21-dimensional particle features which we use for the classification stage.

We also used the pre-trained vectors trained on part of Google News dataset (about 100 billion words). The model contains 300-dimensional vectors for 3 million words and phrases. Also, we used 300-dimensional pre-trained vectors trained with GloVe. GloVe (Pennington et al., 2014) is an unsupervised learning algorithm for obtaining vector representations for words. Simply put, GloVe allows us to take a corpus of text, and intuitively transform each word in that corpus into a position in a high-dimensional space.

Sense Class	#VPCs in Sense Class		
	Train	Verification	Test
Vert-up	24	33	27
Goal-up Cmpl-up	21	24	25
Refl-up	15	3	8

Table 1: Frequency of items in each class for the 3-way task

Sense Class	#VPCs in Sense Class		
	Train	Verification	Test
Vert-up	24	33	27
Goal-up Cmpl-up Refl-up	36	27	33

Table 2: Frequency of items in each class for the 2-way task

Now that we have all of our features, we decide to utilize each set of features alone and with the combinations of the particle features for the classification step. For each set of features, we concatenate the particle features at the end of their vectors and get 321-dimensional feature space. Then we decide to test whether adding the features gathered from the base verb, the base verb as part of the VPC, and the base verb as part of the VPC with the ”up” as its particle will improve the result or not. We also add particle features just like before and as a result, we have 921-dimensional feature space to work with.

We take the sense classes of our base paper (Cook and Stevenson, 2006) for items we have. They generally define 4 classes for the particle up. Since we wanted to compare our results with their work, we also re-implement the 2-way and 3-way classification tasks to see how our features perform on these sense classes. Table 1 and Table 2 show the distribution of senses in each dataset in the 3-way task and 2-way task, respectively.

For classification, we use 3 main classifiers. We use linear SVM (SVC) implementation of (Pedregosa et al., 2011). Also, we do the classification task by taking the weighted version of each classifier in addition to eliminating the effects of the variation in class size on the classifiers. As our second classifier, we used SVM with the kernel type of a radial basis function, implementation of (Pedregosa et al., 2011) in python, same as (Cook and Stevenson, 2006). In this way, we had to set two additional parameters, cost and gamma. We set the input parameters, cost and gamma, using 10-fold cross-validation on the training data and using the same method as (Joanis et al., 2008), we perform a grid search through possible values of these parameters to find the optimal combination. The values we considered for our cost and gamma were the same as the work of (Cook and Stevenson, 2006) and for each combination, we do 10-fold cross-validation on the training data using 10 random restarts. Since we want to eliminate the effect of the class size on the classifier we do the classification task on the weighted version of this classifier as well. Our last classifier was logistic regression which is a fundamental classification technique and it belongs to the group of linear classifiers.

Our evaluation metric is macro-average recall the same as (Cook and Stevenson, 2006) as they pick the measure that weighs each class evenly. It is as same as average per-class accuracy that gives equal weight to each class.

5 Experimental Results

We present experimental results for both verification data and test data on each set of the features individually and in combination. We run all the experiments on both the 2-way and 3-way sense classification. As we stated before, our features capture the semantic and syntactic properties of verbs and VPCs. We summarized the results in Table 3 and Table 4 and discussed in turn below.

Features	2-way task Verification Data					
	SVC	W - SVC	SVC (kernel=rbf)	W - SVC (kernel=rbf)	Logistic Regression	W - Logistic Regression
BNC - Word2vec (CW)	55%	55%	57%	50%	51%	51%
BNC - Word2vec (CW) + PF	56%	56%	57%	50%	49%	51%
BNC - Word2vec (SG)	52%	52%	54%	58%	47%	53%
BNC - Word2vec (SG) + PF	54%	54%	54%	58%	49%	51%
BNC - Word2vec (CW_V)	55%	56%	52%	53%	59%	56%
BNC - Word2vec (CW_V) + PF	55%	55%	55%	51%	59%	56%
BNC - Word2vec (SG_V)	56%	56%	55%	53%	56%	53%
BNC - Word2vec (SG_V) + PF	56%	56%	52%	55%	57%	56%
BNC - Word2vec (CW_VPC)	51%	51%	53%	51%	53%	53%
BNC - Word2vec (CW_VPC) + PF	53%	50%	55%	59%	54%	56%
BNC - Word2vec (SG_VPC)	58%	**61**%	51%	59%	56%	52%
BNC - Word2vec (SG_VPC) + PF	**60**%	**61**%	51%	52%	58%	53%
BNC - Word2vec (CW_VPC_UP)	**60**%	**60**%	**61**%	55%	**61**%	**61**%
BNC - Word2vec (CW_VPC_UP) + PF	**60**%	**60**%	**63**%	59%	**60**%	57%
BNC - Word2vec (SG_VPC_UP)	55%	55%	56%	57%	59%	55%
BNC - Word2vec (SG_VPC_UP) + PF	52%	57%	56%	**60**%	54%	56%
Google - Word2vec	51%	51%	55%	55%	54%	54%
Google - Word2vec + PF	56%	56%	57%	56%	54%	57%
GloVe (glove_big)	59%	59%	**64**%	**60**%	57%	58%
GloVe(glove_big) + PF	58%	58%	**63**%	**62**%	55%	**61**%
Particle Features	50%	52%	56%	57%	57%	58%
BNC - CW_V_VPC_VPC(UP) 900D	58%	58%	58%	50%	59%	**61**%
BNC - CW_V_VPC_VPC(UP)_PF 921D	58%	58%	58%	50%	58%	**62**%
BNC - SG_V_VPC_VPC(UP) 900D	51%	51%	51%	**61**%	51%	52%
BNC - SG_V_VPC_VPC(UP)_PF 921D	51%	51%	53%	59%	51%	54%

Table 3: Macro-Average Recall using linguistic features for Verification Data in 2-way task

We present our experimental results for both the 3-way and the 2-way task on verification data and test data on each set of the features, individually and in combination.

The best performance across the datasets is attained using the features extracted by training word2vec on the British National Corpus dataset in both the 2-way task and the 3-way task on verification data and unseen test data. The result for the verification data for the 2-way task can be seen in Table 3. In the work of (Cook and Stevenson, 2006) they did not use word2vec and they manually created the slot features. Since the word embeddings capture the same information as the slot features do (Chen et al., 2013), we can compare the result from the two sets of features against each other. The best result we get is when we train the word2vec on the words from the British National Corpus and we add the symbol "_VPCUP" to the verbs that are part of the VPC and take "up" as their particle. In this way we wanted to let the word2vec to better capture the VPCs we are looking for. Also, adding the particle features to the word embedding feature improves the result slightly, however, these findings are not consistent in all our experiments. We think that using bigger datasets and develop more detailed sets of features that better capture the information that the particle features aim to capture can be considered as a future work.

The macro-average recalls we achieve with the linguistic features correspond to 5-7% improvements in the 2-way task as you can see in Table 3 and Table 4 and 12-15% improvements in the 3-way task. We can conclude that simply using the word embedding features trained on the BNC for classifying the particle semantics can give us acceptable results. An interesting point to note is that mostly the CBOW version of word2vec gives us better results than the SkipGram version through our experiment. Investigating the reason of this phenomena is not in the scope of this paper. We can see that concatenating all the features (i.e. verb+"_V", verb+"_VPC", verb+"_VPCUP") does not affect on the result and it shows that these information are not complementary. Amongst the classifiers, mostly the weighted version of the SVM with the radial basis function as its kernel type gave us a better result. Our results are mostly respectable to the power of the word embedding models in order to capture the semantic and syntactic features. The google pre-trained word embedding does not perform well in our experiment but glove pre-trained word embedding with 300-dimensional vectors perform interestingly good generally. The re-implemented particle features from (Cook and Stevenson, 2006) work as well as they reported to their paper however we get some inconsistent improvements with our implementation. We hypothesize that the combination of the particle features with the word embedding features will give us an increase in the

	2-way task					
	Test Data					
Features	SVC	W - SVC	SVC (kernel=rbf)	W - SVC (kernel=rbf)	Logistic Regression	W - Logistic Regression
BNC - Word2vec (CW)	48%	48%	53%	50%	49%	49%
BNC - Word2vec (CW) + PF	49%	49%	53%	50%	49%	49%
BNC - Word2vec (SG)	47%	47%	50%	49%	48%	48%
BNC - Word2vec (SG) + PF	48%	48%	48%	49%	50%	47%
BNC - Word2vec (CW_V)	44%	44%	49%	52%	49%	46%
BNC - Word2vec (CW_V) + PF	44%	44%	44%	52%	48%	46%
BNC - Word2vec (SG_V)	49%	49%	45%	50%	50%	50%
BNC - Word2vec (SG_V) + PF	48%	48%	45%	50%	50%	50%
BNC - Word2vec (CW_VPC)	55%	52%	**57%**	**57%**	53%	53%
BNC - Word2vec (CW_VPC) + PF	55%	53%	56%	**60%**	50%	53%
BNC - Word2vec (SG_VPC)	50%	50%	52%	56%	56%	**60%**
BNC - Word2vec (SG_VPC) + PF	54%	55%	51%	55%	56%	56%
BNC - Word2vec (CW_VPC_UP)	55%	55%	56%	**61%**	54%	56%
BNC - Word2vec (CW_VPC_UP) + PF	**61%**	**61%**	**64%**	**60%**	**61%**	**58%**
BNC - Word2vec (SG_VPC_UP)	**58%**	**58%**	**57%**	52%	55%	53%
BNC - Word2vec (SG_VPC_UP) + PF	56%	55%	**60%**	52%	55%	56%
Google - Word2vec	41%	41%	41%	42%	40%	40%
Google - Word2vec + PF	43%	43%	40%	40%	40%	36%
GloVe (glove_big)	42%	42%	39%	43%	41%	46%
GloVe(glove_big) + PF	42%	42%	41%	43%	43%	46%
Particle Features	50%	54%	54%	52%	**64%**	**66%**
BNC - CW_V_VPC_VPC(UP) 900D	50%	50%	52%	50%	40%	50%
BNC - CW_V_VPC_VPC(UP)_PF 921D	50%	50%	52%	50%	46%	50%
BNC - SG_V_VPC_VPC(UP) 900D	47%	47%	47%	49%	47%	49%
BNC - SG_V_VPC_VPC(UP)_PF 921D	47%	47%	49%	49%	47%	50%

Table 4: Macro-Average Recall using linguistic features for Test Data in 2-way task

performance over either set of linguistic features used individually. We conclude that the combination does indeed give us better results we can further analyze more features to capture the information related to particles to get a more consistent improvement over data in future works.

6 Conclusion

A lot of work has been recently done in the area of assessing the compositionality of VPCs but the did not focus on a way to determine the particular meaning of the components. In this paper, we focus on the semantic contribution of the particle. We used a set of 180 VPCs annotated by (Cook and Stevenson, 2006) according to the sense class of "up". We develop features that capture the semantic and syntactic information of the VPCs and also we re-implement the particle features developed by the baseline paper we had (Cook and Stevenson, 2006). We show that using the British National Corpus to train our word embedding model, word2vec,adding a symbol to the verbs that are part of the VPC and have "up" as their particle, we outperform the baseline at least 7%. We demonstrate that adding particle features that are from the work of (Cook and Stevenson, 2006) improve the results in general but not consistently. For the future work, since BNC is not consider as a big dataset, we can extract our features from bigger dataset like Wikipedia and re-implement the features. A broader range of features can be developed from the base to better capture the semantics of the particles in the VPCs. Also we can develop more annotated VPCs to better test our hypothesize. We can also consider methods for token-based semantic determination.

References

Yukiko Sasaki Alam. 2004. Decision trees for sense disambiguation of prepositions: Case of over. In *Proceedings of the Computational Lexical Semantics Workshop at HLT-NAACL 2004*, pages 52–59.

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

Lou Burnard. 1995. *British National Corpus: Users Reference Guide British National Corpus Version 1.0.* Oxford Univ. Computing Service.

Yanqing Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2013. The expressive power of word embeddings. *arXiv preprint arXiv:1301.3226*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

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

Ann Copestake, Fabre Lambeau, Aline Villavicencio, Ivan Sag, Francis Bond, Timothy Baldwin, and Dan Flickinger. 2002. Multiword expressions: Linguistic precision and reusability.

Bruce Fraser. 1974. The phrasal verb in english.

Eric Joanis, Suzanne Stevenson, and David James. 2008. A general feature space for automatic verb classification. *Natural Language Engineering*, 14(3):337–367.

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*, pages 73–80. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Navnita Nandakumar, Bahar Salehi, and Timothy Baldwin. 2018. A comparative study of embedding models in predicting the compositionality of multiword expressions. In *Proceedings of the Australasian Language Technology Association Workshop 2018*, pages 71–76, Dunedin, New Zealand, December.

Navnita Nandakumar, Timothy Baldwin, and Bahar Salehi. 2019. How well do embedding models capture non-compositionality? a view from multiword expressions. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 27–34.

Tom O'Hara and Janyce Wiebe. 2003. Preposition semantic classification via penn treebank and framenet. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 79–86. Association for Computational Linguistics.

Jon Patrick and Jeremy Fletcher. 2005. Classifying verbparticle constructions by verb arguments. In *Proceedings of the Second ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their use in Computational Linguistics Formalisms and Applications*, pages 200–209.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Siva Reddy, Diana McCarthy, and Suresh Manandhar. 2011. An empirical study on compositionality in compound nouns. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 210–218.

Hazem Al Saied, Marie Candito, and Mathieu Constant. 2019. Comparing linear and neural models for competitive MWE identification. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 86–96, Turku, Finland, September–October. Linköping University Electronic Press.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2015. A word embedding approach to predicting the compositionality of multiword expressions. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 977–983, Denver, Colorado, May–June. Association for Computational Linguistics.

Giancarlo Salton, Robert J Ross, and John Kelleher. 2016. Idiom token classification using sentential distributed semantics.

Nathan Schneider, Dirk Hovy, Anders Johannsen, and Marine Carpuat. 2016. SemEval-2016 task 10: Detecting minimal semantic units and their meanings (DiMSUM). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 546–559, San Diego, California, June. Association for Computational Linguistics.

Duyu Tang, Furu Wei, Bing Qin, Nan Yang, Ting Liu, and Ming Zhou. 2015. Sentiment embeddings with applications to sentiment analysis. *IEEE transactions on knowledge and data Engineering*, 28(2):496–509.

Susi Wurmbrand. 2000. The structure (s) of particle verbs. *Ms., McGill University*.

Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398.