Convolution Kernels for Subjectivity Detection

Michael Wiegand and **Dietrich Klakow**
Spoken Language Systems
Saarland University
D-66123 Saarbrücken, Germany
{Michael.Wiegand|Dietrich.Klakow}@lsv.uni-saarland.de

Abstract

In this paper, we explore different linguistic structures encoded as convolution kernels for the detection of subjective expressions. The advantage of convolution kernels is that complex structures can be directly provided to a classifier without deriving explicit features. The feature design for the detection of subjective expressions is fairly difficult and there currently exists no commonly accepted feature set. We consider various structures, such as constituency parse structures, dependency parse structures, and predicate-argument structures. In order to generalize from lexical information, we additionally augment these structures with clustering information and the task-specific knowledge of subjective words. The convolution kernels will be compared with a standard vector kernel.

1 Introduction

One of the most prominent subtasks in sentiment analysis is the detection of subjectivity. Given some expression in a particular context, one is to decide whether this expression conveys some subjective meaning:

1. The United States and its principal allies have acted with **exceptional** caution over recent weeks to the **unbelievable provocation** of the 11 September attacks.

2. General Lucas Rincon held a brief news conference to say Mr Chavez' resignation had been **demanded** and **accepted**.

Though there are several words, such as *horrible* or *ambitious*, that are subjective a priori, there are many ambiguous expressions being subjective only in particular contexts, such as the word *reactions* which is subjective in Sentence 3 but is not subjective in Sentence 4. These examples also illustrate that the contextual information is very helpful in order to decide whether a target word is subjective or not. In Sentence 3, the words that are syntactically related to the target word, e.g. the modifier *healthy* or its governing predicate *provokes*, are particularly predictive as they can be considered subjective expressions themselves.

3. That is a bitter pill to swallow in a thoroughly non-militaristic society such as ours, where the clash of weapons provokes healthy **reactions** of repulsion.

4. Although computers with DNA input and output have been made before, they have always involved a laborious series of *reactions*, each requiring human supervision.

In this paper, we explore different linguistic structures encoded as convolution kernels for the detection of such subjective expressions. We assume that contextual information of structures other than lexical units is useful for this task. The advantage of convolution kernels is that complex structures can be directly provided to a classifier without deriving explicit features. The feature design for the detection of subjective expressions is fairly difficult and there currently exists no commonly accepted feature set. Therefore, we assume that the usage of convolution kernels for this task may be suitable.

We consider various linguistic levels of representation commonly used for classification and extraction tasks in natural language processing, such as constituency parse structures, dependency parse structures, and predicate-argument structures. In order to generalize from lexical information, we additionally augment these structures with clustering information and the task-specific knowledge of

subjective words. The convolution kernels will be compared with a standard vector kernel.

2 Related Work

Convolution kernels have been shown to be effective in various tasks in natural language processing, ranging from relation extraction (Bunescu and Mooney, 2005; Zhang et al., 2006; Nguyen et al., 2009), semantic role labeling (Moschitti et al., 2008) to question answering (Zhang and Lee, 2003; Moschitti, 2008). In sentiment analysis, this method has been successfully applied on opinion holder extraction (Wiegand and Klakow, 2010).

There is no general agreement as to whether linguistic information is useful for text classification tasks in sentiment analysis (which next to subjectivity detection also comprises polarity classification[1]). For supervised document-level analysis, traditional word-level features (i.e. bag of words/n-grams) are usually sufficient (Ng et al., 2006). The usage of more expressive features has been found more effective on fine-grained sentiment analysis, in particular, the classification at word/phrase level (Wilson et al., 2005; Karlgren et al., 2010; Johansson and Moschitti, 2010). The features used in those works have been manually designed and comprise various levels of representation, such as grammatical relations or predicate-argument structures. Johansson and Moschitti (2010) also use a tree kernel encoding dependency parse trees, however, there is no significant improvement achieved by that structure.

In this work, we not only consider dependency parse trees for convolution kernels but also other linguistic levels of representation. Moreover, we also consider appropriate substructures rather than the structures derived from an entire sentence. The latter approach has already been proved effective in opinion holder extraction (Wiegand and Klakow, 2010).

3 Data

As a labeled corpus, we use the Multi-Perspective Question Answering Corpus (MPQA-2.0)[2] which is a widely used corpus annotated with fine-grained information, such as expression-level subjectivity annotation. As subjective expressions, we consider nouns, (full) verbs, and adjectives being labeled as a *direct subjective* or *expressive subjectivity*. We excluded those expressions with *low* or *implicit* subjectivity.[3] Please note that the expressions that are labeled as subjective need not be individual words but can also be phrases. For our approach, we will then consider each word of this phrase being either a noun, verb, or adjective separately, e.g. for Sentence 5 we will consider *biggest* and *story* as two subjective words.

5. You're making us leave as the **biggest story** gets here.

4 Method

4.1 Support Vector Machines and Kernel Methods

Support Vector Machines (SVMs) are one of the most robust supervised machine learning techniques in which training data instances $\vec{x}$ are separated by a hyperplane $H(\vec{x}) = \vec{w} \cdot \vec{x} + b = 0$ where $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$ (Joachims, 1999). The hyperplane can be reformulated with a kernel function $K : X \times X \rightarrow \mathbb{R}$ that computes the similarity of two data instances $\vec{x}_i$ and $\vec{x}_j$ ($\vec{x}_i \wedge \vec{x}_j \in X$). While in a standard vector kernel these instances are represented by manually defined features, such as bag of words, in a convolution kernel the instances can be described by more complex discrete structures, such as trees or sequences. A convolution kernel function is an algorithm that specifies how the similarity of two instances represented by such discrete structures can be computed.

The convolution kernels we evaluate in this work are two tree kernels: Subset Tree Kernel (STK) (Collins and Duffy, 2002) and Partial Tree Kernel (PTK_{basic}) (Moschitti, 2006). We will focus exclusively on tree structures since they largely outperform other kernels, such as sequence kernels (Wiegand and Klakow, 2010). In a tree kernel, the similarity of two trees is computed by counting the number of common tree fragments. In STK, a tree fragment can be any set of nodes and edges of the original tree provided that every node has either all or none of its children. This constraint makes that kind of kernel well-suited for constituency trees (Zhang et al., 2006; Nguyen et al., 2009; Wiegand and Klakow, 2010) that have

[1] Polarity classification is the task of distinguishing between positive and negative opinions.

[2] www.cs.pitt.edu/mpqa/databaserelease

[3] Beyond the experiments presented in this paper, we also experimented with other definitions, e.g. also including *low* subjective expressions. Though these definitions usually result in different absolute numbers in the evaluation, the relation between the different methods remains similar.

been generated by context free grammars since the constraint corresponds to the restriction that no grammatical rule must be broken. For example, STK enforces that a subtree, such as *[VP [VBZ, NP]]*, cannot be matched with *[VP [VBZ]]* since the latter *VP* node only possesses one of the children of the former.
PTK_{basic} is more flexible since the constraint of STK on nodes is relaxed. This makes this type of tree kernel less suitable for constituency trees. We, therefore, apply it only to trees representing dependency trees (thus following Johansson and Moschitti (2010)) and predicate-argument structures (thus following Moschitti (2008)).

4.2 Bag of Words

The simplest form of contextual information to be used for our task is lexical information, i.e. bag of words. Even though it is extremely simple and cheap to produce, this level of representation is known to be fairly robust for different kinds of text classification tasks. We will examine several context windows and encode this representation in a standard vector kernel.

4.3 Convolution Kernels

The following subsections present the different kinds of tree structures that are used for convolution kernels.

4.3.1 Constituency Parse Structures (CON)

Wiegand and Klakow (2010) showed for opinion holder extraction that using the entire constituency parse tree of a sentence produces very low performing classifiers. Structures derived from an entire sentence contain too much irrelevant information for such a task at expression level. We assume that the same is true for the detection of subjectivity. Wiegand and Klakow (2010) use subtrees derived from scopes. The best performing subtree is the tree with the predicate scope, i.e. a subtree with the boundaries being the candidate or target expression and the nearest predicate. We also assume that this structure is meaningful for our task. As already discovered in previous work on the detection of subjective expressions (Riloff and Wiebe, 2003; Riloff and Wiebe, 2003; Wilson et al., 2005), discriminant patterns often encode a relation between the target expression and the nearest predicate. An illustration of this substructure is given in Figure 1(a). We use the Stanford Parser (Klein and Manning, 2003) for obtaining constituency parse trees.

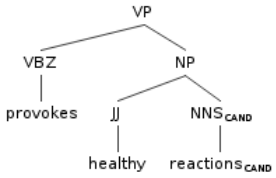
4.3.2 Dependency Parse Structures (DEP)

Apart from manually designed features, Johansson and Moschitti (2010) also test a tree kernel for subjectivity detection using a dependency parse. However, the entire parse comprising a sentence is considered. The resulting tree kernel does not show any significant improvement (again presumably because of the large amount of irrelevant information). The usefulness of particular (usually direct) relations, however, has been found effective on other related tasks in sentiment analysis (Jakob and Gurevych, 2010; Wiegand and Klakow, 2010). We therefore only consider the subtree exclusively containing the lexical units that are connected to the target word by a direct syntactic dependency relationship (i.e. direct parent and direct children). The precise encoding of the pertaining information (i.e. part-of-speech, grammatical relation, and lexical information) in the resulting tree is taken from (Johansson and Moschitti, 2010). An illustration of this substructure is given in Figure 1(b). Again, we use the Stanford Parser (Klein and Manning, 2003) for obtaining dependency parse trees.

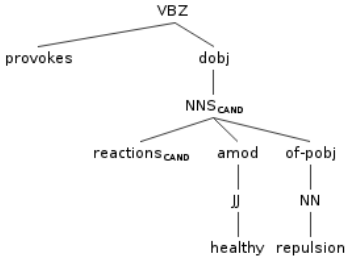
4.3.3 Predicate-Argument Structures (PAS)

Predicate-argument structures (PAS), in particular, semantic role labeling has been shown to be effective for many information extraction tasks, including opinion holder extraction (Kim and Hovy, 2006; Wiegand and Klakow, 2010) and opinion target extraction (Kim and Hovy, 2006). Johansson and Moschitti (2010) also examine this level of representation for subjectivity detection. However, they employ manual features derived from these structures rather than using a corresponding tree kernel.
We follow Wiegand and Klakow (2010) for the encoding of these structures as tree kernels, that is we restrict ourselves to structures in which the target word is either a predicate or some argument. We derive our predicate-argument structures from a semantic parse based on the PropBank annotation scheme (Kingsbury and Palmer, 2002). Semantic roles are obtained by using the parser by Zhang et al. (2008). A special property of PAS is that a data instance, i.e. the information regarding one target word and its particular context, is represented by a set of those structures

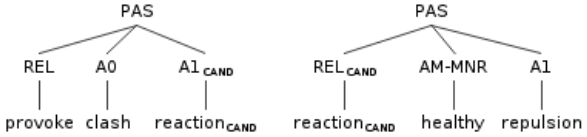
rather than a single structure. Thus, the actual partial tree kernel function we use for this task, PTK, sums over all possible pairs PAS_l and PAS_m of two data instances x_i and x_j: $PTK(x_i, x_j) = \sum_{PAS_l \in x_i} \sum_{PAS_m \in x_j} PTK_{basic}(PAS_l, PAS_m)$. An illustration of these substructures is given in Figure 1(c).



(a) CON



(b) DEP



(c) PAS

Figure 1: Illustration of the different tree structures employed for convolution kernels derived from Sentence 3 with *reactions* as the target word.

4.3.4 Augmentation with Clustering

A common type of unsupervised generalization is clustering. Words which co-occur with each other are automatically grouped into clusters. Ideally, a cluster thus contains words with similar syntactic/semantic properties. The cluster membership of individual words is induced from a large unlabeled corpus.

As the context windows of our target expressions contain fairly sparse lexical information, the additional usage of the cluster membership might be useful. Turian et al. (2010) have shown that for named-entity recognition, i.e. a task which faces similar lexical sparseness, features based on such a cluster membership improve the overall performance. For clustering, we chose Brown clustering (Brown et al., 1992) which is the best performing algorithm in (Turian et al., 2010). This algorithm induces clusters with the help of co-occurrence statistics of bigrams. We augment our structures with the clustering information. We add the node with a cluster label in such a way that it directly dominates the pertaining lexical node.

As a software we use SRILM (Stolcke, 2002) with the default algorithm. The clusters are induced on the North American News Text Corpus (LDC95T21). We chose this corpus as it contains news texts similar to our evaluation corpus (i.e. MPQA). Following Turian et al. (2010), we induced 1000 clusters.

As many names of persons and organizations can be very domain-specific, they may not appear in the corpus from which clusters are induced. Consequently, these expressions cannot be assigned to a cluster. We try to compensate this by incorporating the knowledge about named entities in tree kernels, i.e. instead of assigning some expression to a cluster we assign it to a named entity type. Named-entity information is obtained by the Stanford tagger (Finkel et al., 2005).

4.4 The Different Settings

We want to examine the behavior of the different kernel types under different circumstances. The first setting $NoTW$ considers a prediction for unseen target words. We assume that having observed a particular target word frequently in the training data (in particular if it is fairly unambiguous) makes it easy for the classification when it is observed as a test instance, i.e. due to the prior knowledge about the lexical unit the consideration of context is less critical. If a word, however, has not been observed in the training data the consideration of context becomes much more important. For instance, an unknown word that is modified by an intensifier *extremely* is more likely to contain a subjective meaning (as in *extremely* ***nice***) than a word that is not modified by such an expression. The question arises whether in these cases structural context has an even larger impact than lexical context. For reasons of simplicity, we simulate unseen target words by discarding the vector kernel features indicating the lexical unit of the target word and replacing the label of the corresponding leaf node in the tree kernels by a generic symbol.

The second setting TW is the main setting we use for most experiments. In this setting, the target

word is considered, i.e. we have dedicated features (original word form + lemma[4]) in the vector kernel and we retain the label of the leaf node representing the target word in the tree kernels.
The final setting TW+OP also incorporates the knowledge about subjective expressions. For the vector kernel, we add features indicating whether the target word is either a weak or strong subjective expression. Moreover, we include two features indicating whether the context (i.e. the words surrounding the target word in its context window) contains either at least one weak or strong subjective expression. The knowledge about subjective expressions is drawn from a given sentiment lexicon. We use the Subjectivity Lexicon from the MPQA project (Wilson et al., 2005). Table 1 summarizes the different settings with regard to the respective features in the vector kernel and the encoding of the convolution kernels. An illustration of the different settings on a constituency parse tree (CON) is displayed in Figure 2.

5 Experiments

We used 400 documents of the MPQA corpus for five-fold cross-validation and 133 documents as a development set. We will exclusively report the results that are averages over cross-validation. We report statistical significance on the basis of a paired t-test using 0.05 as the significance level.

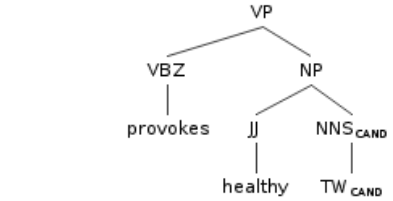
As there is a heavy class imbalance between positive and negative data (e.g. for nouns, more than 85% of the data instances are negative examples), we evaluate with F-Score rather than accuracy. In order to achieve an optimal F-Score, we tuned the cost parameter j (Morik et al., 1999) on the development set using our baseline classifier, i.e. VK.[5] This cost parameter allows to specify a higher penalty on false negative errors. Thus, on data sets with an imbalanced class distribution better models assigning less conservatively labels of the minority class can be produced. We set $j = 4$.

As far as tree kernels are concerned, we used the parameter settings from Moschitti (2008). Kernels were combined using plain summation.

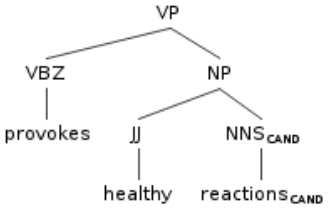
For all experiments apart from those presented in Section 5.3 will be based on our main setting, i.e. TW (see Section 4.4).

[4]The lemma feature is important for normalizing inflectional forms. From the parts of speech that we consider verbs mostly benefit from such a normalization.

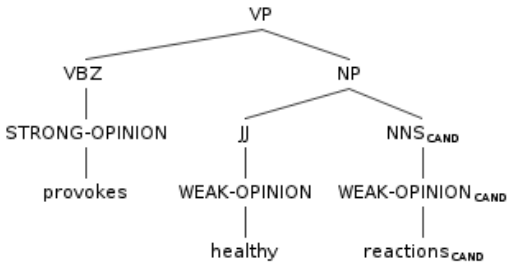
[5]Note that the cost parameter is the only parameter tuned on the development set.



(a) NoTW



(b) TW



(c) TW+OP

Figure 2: Illustration of the different settings (as defined in Table 1) on a constituency parse tree (CON).

5.1 Different Vector Kernel Configurations

Table 2 displays the performance (F-Score) of the vector kernel with different configurations. We vary the window size[6] and the kernel type, i.e. linear or polynomial kernel. We use a second degree polynomial kernel as we obtained the best performance with that type. The table shows that, usually the polynomial kernel always outperforms the linear kernel. Moreover, the best window size is between 5 and 10. The difference between those two window sizes using a polynomial kernel is, however, never statistically significant.

5.2 Different Convolution Kernel Configurations

Table 3 displays the performance of the different convolution kernels. As far as individual kernels are concerned, CON and DEP are fairly similar, PAS performs much worse. Overall, the best

[6]Window size n means that n words both preceding and following the target word are taken into consideration.

Setting	Vector Kernel (Features)	Convolution Kernel (Encoding)
NoTW	context words within window	label of target word node is replaced by generic *TW*
TW ***(default)***	*all features from NoTW +* target word lemma of target word	label of target word node is not changed
TW+OP	*all features from TW +* is target word a weak/strong subjective word? has context more than one weak/strong subjective word?	label of target word node is not changed augment trees if weak/strong subjective words are present

Table 1: The different settings with the design of the corresponding kernel types.

	Noun		Verb		Adj	
WS	**linear**	**poly**	**linear**	**poly**	**linear**	**poly**
2	53.66	54.77	59.05	62.34	63.23	64.51
5	**55.90**	**59.29**	61.58	63.63	64.16	**66.52**
10	55.68	58.87	62.34	**64.46**	**64.52**	66.38
20	54.76	55.90	**62.68**	63.42	62.79	64.33

Table 2: F-Score of vector kernels using different window sizes (WS) and kernel types (using default setting TW).

combination is CON and DEP. A combination of PAS with the other convolution kernels only has a notable positive impact for verbs, which does not come as a surprise since semantic role labeling is clearly verb-centred (still, it is too sparse to be used as a kernel on its own). With the exception of verbs, a combination of all convolution kernels is not necessary, i.e. the combination of CON and DEP suffices (there is no significant improvement achieved by also adding PAS).

We were surprised that in spite of the fact that PAS as an individual kernel performs very poorly, it never harms performance when it is added to another kernel type and occasionally also causes some improvement. We ran some more experiments and found that the low performance of PAS is mainly due to the fact that a default value for the depth parameter μ (Zhang and Lee, 2003) was used. With an increased depth parameter (e.g. $\mu = 0.8$) which more suitably accounts for the low depth of flat predicate-argument structures, PAS performs much better. However, since for combined structures, the optimized parameter has only a marginal impact and we only use a default μ for all other configurations, for reasons of consistency we just use PAS with default settings.

Finally, the usage of tree augmentation using clustering (i.e. *best aug*), is always beneficial. Table 4 lists the content of three clusters that are induced. All three of them mostly contain expressions one would intuitively label as subjective expressions. This means that to a certain extent, clustering is able to group subjective expressions and objective expressions into different clusters.

We also ran some experiments using different structures than those presented in Section 4.3. Though for opinion holder extraction, a combination of scopes is useful for the constituency parse trees (Wiegand and Klakow, 2010), we found that for this task adding other scopes does not help. Further extensions of the dependency structure, e.g. by also including indirect syntactic dependency relationships (grandparent and grandchildren of the target word) did not help either. These results suggest that for this task there is little room for improvement by applying state-of-the-art convolution tree kernels on other structures than the one presented in this paper.

5.3 Comparing Vector Kernels with Convolution Kernels

Table 5 compares the best vector kernel (VK) for each respective part of speech and the combination of the best convolution kernel (CKs) with a vector kernel. As from the results for the individual vector kernels on the setting TW (Table 2) it is not obvious which window size is best (i.e. 5 or 10), we always tested both sizes and will report the best result in Table 5. For the combination with CKs, we always chose a vector kernel with the window size of 10 as, unlike the experiments on VK, we observed that this window size was consistently better than using the window size of 5.

We examine this combination on all three set-

	Noun			Verb			Adj		
Kernels	**Prec**	**Rec**	**F**	**Prec**	**Rec**	**F**	**Prec**	**Rec**	**F**
CON	53.49	52.23	52.84	47.15	68.89	55.98	57.07	67.49	61.83
DEP	46.43	59.57	52.17	43.68	79.94	56.49	49.72	74.78	59.72
PAS	21.12	39.39	12.04	**54.91**	12.08	18.56	29.97	49.51	29.87
CON+DEP	**52.81**	57.84	55.23	49.01	74.41	59.10	57.06	72.40	63.80
CON+PAS	52.00	56.94	54.34	48.90	78.25	60.18	56.88	68.19	62.01
DEP+PAS	47.78	59.85	53.12	45.89	81.14	58.76	50.25	73.44	59.66
all	52.23	59.57	55.64	49.60	77.75	60.65	**57.09**	71.96	63.65
best aug	52.60	**61.90**	**56.80**	50.30	77.99	**61.16**	56.74	**75.25**	**64.69**

Table 3: Performance of the different convolution kernels (using default setting TW).

tings, i.e. $NoTW$, TW and TW+OP (see Section 4.4). The table shows that the overall performance of $NoTW$ is worst and that of TW+OP is best, which is quite intuitive as it corresponds to the amount of knowledge encoded in the configurations. The gap between $NoTW$ and TW is the largest indicating that the knowledge of the target word itself is extremely important. The knowledge of subjective expressions TW+OP helps but the degree of improvement is smaller.[7]

As far as the relation between vector kernels and convolution kernels is concerned, the combination of vector kernel and convolution kernels is mostly beneficial. With the exception of verbs, we always obtain a significant improvement over just using the best vector kernel.

If we compare the relation between vector kernel and the combination of vector kernel and convolution kernels across the different settings, we also observe that the less is known about the target word, the more helpful the information is that can be drawn from the convolution kernels. For example, when CKs are added to VK on nouns, there is an improvement of 5 percentage points while on the other settings the improvement is usually less than 2 percentage points. In other words, if we have test data in which many target words are unknown, then the structural context information is much more important than if the target words have mostly appeared in the training data.

[7]Since we expected a greater improvement by adding knowledge of subjective words, we ran some more exploratory experiments. We found that if we run SVMs with a default configuration that assigns the equal cost to the different classes , i.e. setting the cost parameter $j = 1$, the relative improvement from TW to TW+OP is considerably larger (though in absolute numbers, the F-Scores are much lower than by using the optimized parameter).

acclaim, admiration, backing, benefit, contempt, disdain, disregard, disrespect, doom, finesse, gratitude, harm, praise, redress, refrain, rent, respect, ridicule, support, sympathy
anticipate, attribute, believe, confess, detest, expect, imply, indicate, liken, mean, owe, presume, pretend, resent, suggest, swear
decent, fantastic, good, handy, lousy, mediocre, miserable, nice, nostalgic, pleasant, shrewd, smart, terrific, wise, wonderful

Table 4: Some automatically induced clusters.

6 Conclusion

In this paper, we examined the usage of convolution kernels for the detection of subjective expressions and compared it to the performance of a vector kernel trained on bag-of-words features. The polynomial vector kernel is a hard baseline. For nouns and adjectives, however, the performance can be significantly increased if in addition to the vector kernel a pair of convolution kernels encoding a constituency parse subtree with predicate scope and a dependency parse subtree encoding the direct relationships between the target word and other words in a sentence is used. Additionally, the trees can be effectively augmented with cluster-membership information.

Acknowledgements

This work was funded by the BMBF project "Software-Cluster" (contract no.: *01IC10S01O) and the German research council DFG through the International Research Training Group. The authors would like to thank Yi Zhang for processing our data with his SRL system and Alessandro Moschitti for providing his toolkit SVMLight-TK.

		Noun			Verb			Adj		
Setting	**Kernels**	**Prec**	**Rec**	**F**	**Prec**	**Rec**	**F**	**Prec**	**Rec**	**F**
NoTW	VK	43.70	60.75	50.85	48.86	76.97	59.77	52.46	73.78	61.32
NoTW	VK+CKs	50.55	63.27	56.19*	48.45	79.56	60.22	54.83	76.80	63.98*
TW	VK	56.33	62.61	59.29	55.21	77.44	64.46	61.70	72.16	66.52
TW	VK+CKs	57.01	65.92	61.12*	53.46	78.90	63.73	61.51	77.21	68.47*
TW+OP	VK	56.01	65.57	60.41	55.59	78.45	65.07	62.62	75.36	68.39
TW+OP	VK+CKs	57.98	67.34	62.29*	55.16	79.36	65.08	62.61	77.93	69.43*

Table 5: Comparison of the (best) vector kernel and convolution kernels using different settings (*: significantly better than VK with $p \leq 0.05$).

References

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

R. C. Bunescu and R. J. Mooney. 2005. Subsequence Kernels for Relation Extraction. In *Proc. of NIPS*.

M. Collins and N. Duffy. 2002. New Ranking Algorithms for Parsing and Tagging. In *Proc. of ACL*.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proc. of ACL*.

N. Jakob and I. Gurevych. 2010. Extracting Opinion Targets in a Single- and Cross-Domain Setting with Conditional Random Fields. In *Proc. of EMNLP*.

T. Joachims. 1999. Making Large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

R. Johansson and A. Moschitti. 2010. Syntactic and Semantic Structure for Opinion Expression Detection. In *Proc. of CoNLL*.

J. Karlgren, G. Eriksson, O. Täckström, and M. Sahlgren. 2010. Between Bags and Trees - Constructional Patterns in Text Used for Attitude Identification. In *Proc. of ECIR*.

S. Kim and E. Hovy. 2006. Extracting Opinions, Opinion Holders, and Topics Expressed in Online News Media Text. In *Proc. of the ACL Workshop on Sentiment and Subjectivity in Text*.

P. Kingsbury and M. Palmer. 2002. From TreeBank to PropBank. In *Proc. of LREC*.

D. Klein and C. D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proc. of ACL*.

K. Morik, P. Brockhausen, and T. Joachims. 1999. Combining statistical learning with a knowledge-based approach - A case study in intensive care monitoring. In *Proc. in ICML*.

A. Moschitti, D. Pighin, and R. Basili. 2008. Tree Kernels for Semantic Role Labeling. *Computational Linguistics*, 34(2):193 – 224.

A. Moschitti. 2006. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In *Proc. of ECML*.

A. Moschitti. 2008. Kernel Methods, Syntax and Semantics for Relational Text Categorization. In *Proc. of CIKM*.

V. Ng, S. Dasgupta, and S. M. Niaz Arifin. 2006. Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews. In *Proc. of COLING/ACL*.

T.-V. T. Nguyen, A. Moschitti, and G. Riccardi. 2009. Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation Extraction. In *Proc. of EMNLP*.

E. Riloff and J. Wiebe. 2003. Learning Extraction Patterns for Recognizing Subjective Expressions. In *Proc. of EMNLP*.

A. Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proc. of ICSLP*.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word Representations: A Simple and General Method for Semi-supervised Learning. In *Proc. of ACL*.

M. Wiegand and D. Klakow. 2010. Convolution Kernels for Opinion Holder Extraction. In *Proc. of HLT/NAACL*.

T. Wilson, J. Wiebe, and P. Hoffmann. 2005. Recognizing Contextual Polarity in Phrase-level Sentiment Analysis. In *Proc. of HLT/EMNLP*.

D. Zhang and W. S. Lee. 2003. Question Classification using Support Vector Machines. In *Proc. of SIGIR*.

M. Zhang, J. Zhang, and J. Su. 2006. Exploring Syntactic Features for Relation Extraction using a Convolution Tree Kernel. In *Proc. of HLT/NAACL*.

Y. Zhang, R. Wang, and H. Uszkoreit. 2008. Hybrid Learning of Dependency Structures from Heterogeneous Linguistic Resources. In *Proc. of CoNLL*.