[LVIC-LIMSI]: Using Syntactic Features and Multi-polarity Words for Sentiment Analysis in Twitter

Morgane Marchand[1,2], **Alexandru Lucian Ginsca**[1], **Romaric Besançon**[1], **Olivier Mesnard**[1]
(1) CEA-LIST, DIASI, LVIC
CEA SACLAY - Nano-INNOV - Bt. 861 - Point courrier 173
91191 Gif-sur-Yvette Cedex, France
(2) LIMSI-CNRS
Bat 508, BP133,91403 Orsay Cedex
morgane.marchand@cea.fr; alexandru.ginsca@cea.fr
romaric.besancon@cea.fr; olivier.mesnard@cea.fr

Abstract

This paper presents the contribution of our team at task 2 of SemEval 2013: Sentiment Analysis in Twitter. We submitted a constrained run for each of the two subtasks. In the Contextual Polarity Disambiguation subtask, we use a sentiment lexicon approach combined with polarity shift detection and tree kernel based classifiers. In the Message Polarity Classification subtask, we focus on the influence of domain information on sentiment classification.

1 Introduction

In the past decade, new forms of communication, such as microblogging and text messaging have emerged and became ubiquitous. These short messages are often used to share opinions and sentiments. The *Sentiment Analysis in Twitter* task promotes research that will lead to a better understanding of how sentiment is conveyed in tweets and texts. In this paper, we describe our contribution at task 2 of SemEval 2013 (Wilson et al., 2013). For the *Contextual Polarity Disambiguation* subtask, covered in section 2, we use a system that combines a lexicon based approach to sentiment detection with two types of supervised learning methods, one used for polarity shift identification and one for tweet segment classification in the absence of lexicon words. The third section presents the *Message Polarity Classification* subtask. We focus here on the influence of domain information on sentiment classification by detecting words that change their polarity across domains.

2 Task A: Contextual Polarity Disambiguation

In this section we present our approach for the contextual polarity disambiguation task in which, given a message containing a marked instance of a word or a phrase, the system has to determine whether that instance is positive, negative or neutral in that context. For this task, we submitted a single run using only the tweets provided by the organizers.

2.1 System description

Based on the predominant strategy, sentiment analysis systems can be divided into those that focus on sentiment lexicons together with a set of rules and those that rely on machine learning techniques. For this task, we use a mixed approach in which we first filter the tweets based on the occurrences of words from a sentiment lexicon and then apply different supervised learning methods on the grounds of this initial classification. In Figure 1 we detail the workflow of our system. We use the $+$, $-$ and $*$ symbols to denote a positive, negative and neutral tweet segment, respectively. Also, we use the $a \rightarrow b$ notation when referring to a polarity shift from a to b.

2.1.1 Data preprocessing

The language used in Twitter presents some particularities, such as the use of hashtags or user mentions. In order to maximize the efficiency of language processing methods, such as lemmatization and syntactic parsing, we perform several normalization steps. We remove the # symbol, all @ mentions and links and perform lower case conversion. Also, if a vowel is repeated more than 3 times in a word, we reduce it to

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 418–424, Atlanta, Georgia, June 14-15, 2013.

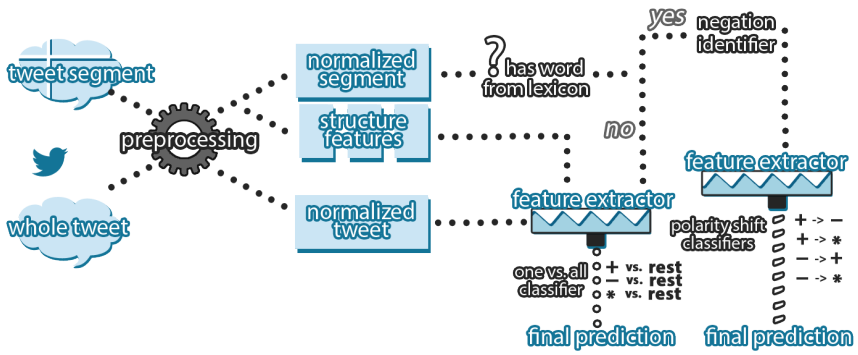

Figure 1: Contextual polarity disambiguation task system description

a single occurrence and we reduce multiple consecutive punctuation marks to a single one. Finally, we lemmatize the normalized text.

Emoticons have been successfully used as sentiment indicators in tweets (Davidov et al., 2010). In our approach, we map a set of positive emoticons to the word *good* and a set of negative emoticons to the word *bad*. We use the following sets of emoticons:

- Positive emoticons: :) , :-) , :D , =) , :') , :o) , :P , >:) , :">, >:|, <3 , ;>, ;) , ;-) , ;>, (: , (;
- Negative emoticons: :(, : (, :-(, :'(, :/ , :<, ;(

Traits of informal language have been used as features in Twitter sentiment classification tasks (Go et al., 2009). In order to avoid the loss of possible useful information, we keep record of the performed normalizations as binary features associated to a tweet segment. We retain the following set of features: *hasPositiveEmoticon, hasNegativeEmoticon, hasHashtag, hasAtSign, hasConsecutivePunctuation, hasConsecutiveVowels, hasUpperCaseWords*.

2.1.2 Classification methods

In a first step, we select tweet segments that contain at least one word from a lexicon and assign to it the polarity of that word. If there are more than one sentiment words with different polarities in the segment, we keep the most frequent polarity and in the few cases where there is an equal number of positive and negative words, we take the polarity of the last one. Next, we look for negation indicators (e.g. *not, 't*) using a set of words and rules and replace them with the *NEG* token. We then identify instances where there is a shift between the polarity predicted from the lexicon and the one from the ground truth. In order to account for the unbalanced datasets(e.g. 192 instances where there is a $+ \rightarrow -$ shift and 3188 where the positive instance was correctly identified from the lexicon) we use cost sensitive classifiers. We define a cost matrix in which the cost of the classifier making a false positive error is three times higher than a false negative error. Using this approach we guide the classifier to provide less but more confident predictions for the existence of a polarity shift while allowing it to make more errors when predicting the absence of a shift. For these classifiers, we use a *Bag of Words* representation of the lemmatized segments. When a word from the sentiment lexicon does not appear in the tweet segment, we use a *one vs. all* classification approach with a SVM classifier and tree kernels. The tree kernel is a function between two trees that computes a normalized similarity score in the range *[0,1]* (Culotta and Sorensen, 2004). For our task, we use an implementation of tree kernels for syntactic parse trees (Moschitti, 2006) that is built on top of the SVM-Light library (Joachims, 1999) in a similar manner to that presented in (Ginsca, 2012). We build the syntactic parse trees with the Stanford CoreNLP library (Klein and Manning, 2003).

2.2 Evaluation and Results

For the experiments presented in this section, we merge the training and development datasets and for the polarity shift and sentiment classification experiments we report the results using a 5-fold cross validation technique over the resulting dataset.

2.2.1 Lexicon choice influence

Considering that the selection of a lexicon plays an important role on the performance of our system, we tested 3 widely used sentiment lexicons: SentiWordNet 3 (Baccianella et al., 2010), Bing Liu's Opinion Lexicon (Hu and Liu, 2004) and MPQA Subjectivity Lexicon (Wilson et al., 2005). Different combinations of these lexicons were tried and in Table 1 we present the top performing ones. Besides the F-Measure for positive (*Fp*) and negative (*Fn*) instances, we also list the percentage of instances in which appears at least one word from the lexicon. SentiWordnet appoints polarity weights to words, ranging from 0 to 1. An important parameter is the threshold over which a word is considered to have a certain polarity. We tested several values (from 0.5 to 0.9 with a step of 0.05) and the best results in terms of F-Measure were obtained for a threshold of 0.75. Our finding is consistent with the value suggested in (Chen et al., 2012).

Lexicon	Found(%)	Fp	Fn
Liu	55.7	0.93	0.85
MPQA	61.4	0.89	0.76
SentWN	79.4	0.86	0.78
Liu+MPQA	67.1	0.89	0.78
Liu+SentWN	79.4	0.87	0.81
Liu+MPQA+SentWN	79.4	0.86	0.81

Table 1: Influence of lexicon on the F-Measure for positive and negative segments

2.2.2 Polarity shift experiments

We tested several classifiers using the Weka toolkit (Hall et al., 2009) and found that the best results were obtained with the Sequential Minimal Optimization (SMO) classifier. For instance, when classifying $+ \rightarrow -$ shifts, SMO correctly identified 91 out of 192 polarity shifts in contrast with 68 and 41 detected by a Random Forests and a Naive Bayes classifier, respectively. For the $+ \rightarrow *$ classification, the SMO classifier finds 2 out of 34 shifts, for $- \rightarrow +$, 15 out of 238 and for $- \rightarrow *$, 2 out of 32 shifts are found. After changing the polarity of sentiment segments as found by the 4 classifiers, we obtain an increase in F-Measure from 0.930 to **0.947** for positive segments and from 0.851 to **0.913** for negative segments. Our choice of the *Bag of Words* model instead of a parse tree representation for these classifiers is justified by the poor performance of tree kernels when dealing with unbalanced data.

2.2.3 Sentiment classification experiments

Model	Class	Avg. F-score
Basic Tree	positive	0.780
	negative	0.645
	neutral	0.227
Tree + Numeric	positive	0.768
	negative	0.590
	neutral	0.132
Tree + Context_2	positive	0.801
	negative	0.676
	neutral	0.231

Table 2: Comparison between different models used for segment polarity classification

In a series of preliminary experiments, we tested several classifiers trained on a *Bag of Words* model and an SVM classifier with a tree kernel. We found that the parse tree representation of a tweet segment provided a higher accuracy. This shows that although small, when a segment contains more than one word, its syntactic structure becomes a relevant feature.
In Table 2 we compare the results of 3 tree based models. In the *Basic Tree* model, we use only the syntactic parse tree representation of a tweet segment. For the *Tree + Numeric* model, we use the initial tree kernel together with a polynomial kernel on the binary structure features presented in section 2.1.1. In the *Tree + Context* model, we include in the parse tree, besides the given section, *k* tokens (words, punctuation) from the whole tweet that surround the selected segment. We performed tests with *k* from *1* to *5* and obtained the best results with a *k* value of 2.

2.2.4 Competition results

For the Twitter dataset, we ranked **4th** out of 23 groups that submitted constrained runs. When combining the results of the constrained and unconstrained submissions, our run was ranked **5th** out of a total of 29 submissions. For the SMS dataset, we ranked **5th** out of a total of 18 groups for the constrained setting and our submission was ranked **5th** out of 24 combined runs. In Table 3, we detail the results we obtained on the competition test datasets.

Class	P	R	F-score
Twitter_positive	0.8623	0.9140	0.8874
Twitter_negative	0.8453	0.8086	0.8265
Twitter_neutral	0.4127	0.1625	0.2332
SMS_positive	0.7107	0.8945	0.7921
SMS_negative	0.8687	0.7609	0.8112
SMS_neutral	0.3684	0.0440	0.0787

Table 3: Competition results overview on the Twitter and SMS datasets

2.3 Discussion

The robustness of our approach is proved by the low standard deviation of the F-Measure scores obtained over each of the the 5 folds used for evaluation (0.026) but also by the small difference between the results we obtained during the development phase and those reported on the competition test dataset. The choice of lexicons results in a trade-off between the percentage of instances classified with either the lexicon and polarity shift or the supervised learning method. Although the first one yields better results and it is apparently desirable to have a better coverage of lexicon terms, this would reduce the number of instances for training a classifier leading to a poorer performance of this approach.

3 Task B: Message Polarity Classification

In this section, we present our approach for the message polarity classification task in which, given a message, the system has to determine whether it expresses a positive, negative, or neutral sentiment. As for Task A, we submitted a single constrained run.

3.1 Preprocessing of the corpora

We use as training corpora the training data, merged with the development data. After the deletion of tweets no longer available, our final training set contains 10402 tweets: 3855 positive, 1633 negative and 4914 objective or neutral. In the preprocessing step, we first remove the web addresses from the tweets to reduce the noise. Then, we extract the emoticons and create new features with the number of occurrences of each type of emoticon. The different emoticons types are presented in Table 4. Then, we lemmatize the text using LIMA, a linguistic analyzer of CEA LIST (Besançon et al., 2010).

:-) :) =) X) x)	Smile
:-(:(=(	Sadness
:-D :D =D X-D XD x-D xD :')	Laugh
;-) ;)	Wink
< 3	Heart
:'-(:'(='(	Tear

Table 4: Common emoticon types

3.2 Boostexter baseline

To classify the tweets, we used the BoosTexter[1] classifier (Schapire and Singer, 2000) in its discrete AdaBoost.MH version, setting the number of iterations to 1000. We used two types of features: a *Bag of Words* of lemmatized uni-, bi- and tri-grams and the number of occurrences of each emoticon type.

Bog of words features	Emoticon type feature
wow lady gaga be great	Smile 1

Table 5: Example of tweet representation

Boostexter is designed to maximize the accuracy, not the F-score, which is the chosen evaluation metric for this task. As the training data contain few negative examples, the classifier tends to under-detect this class. In order to favour the negative class detection, we balance the training corpora. So our final system is trained on 4899 tweets (1633 of each class, chosen randomly). The accuracy results are not presented here. However, the gain between our baseline and our final system has the same order of magnitude.

3.3 Integration of domain information

Some words can change their polarity between two different domains (Navigli, 2012; Yoshida et al., 2011). For example, the word "return" is positive in "I can't wait to return to my book". However, it is often very negative when we are talking about some electronics device, as in "I had to return my phone to the store". This phenomenon happens even in more closely related domains: "I was laughing all the time" is a good point for a comedy film but a bad one for a horror film. We call such words or expressions "multi-polarity words". This phenomenon is different

[1] BoosTexter is a general purpose machine-learning program based on boosting for building a classifier from text.

from polysemy, as a word can keep the same meaning across domains while changing its polarity and it can lead to classification error (Wilson et al., 2009). In (Marchand, 2013), we have shown, on a corpus of reviews, that a sensible amount of multi-polarity words influences the results of common opinion classifiers. Their deletion or their differentiation leads to better classification results. Here, we test this approach on a corpus of tweets.

3.3.1 Domain generation with LDA

In order to apply our method, we need to assign domains to tweets. For that purpose, we use Latent Dirichlet Allocation (LDA) (Blei et al., 2003). We used the Mallet LDA implementation (McCallum, 2002). The framework uses Gibbs sampling to constitute the sample distributions that are exploited for the creation of the topic models. The models are built using the lemmatized tweets from the training and development data. We performed tests with a number of domains ranging from 5 to 25, with a step of 5. Each LDA representation of a tweet is encoded by inferring a domain distribution. For example, if a model with 5 domains is used, we generate a vector of length 5, where each the i-th value is the proportion of terms belonging to the i-th domain.

Domain 1	tonight, watch, time, today
Domain 2	win, vote, obama, black
Domain 3	game, play, win, team
Domain 4	apple, international, sun, anderson
Domain 5	ticket, show, open, live

Table 6: Most representative words of each domain (5 domains version)

In first experiments with crossvalidation on training data, the 5 domains version, presented in Table 6, appears to be the most efficient. Therefore, in the rest of the paper, results are shown only for this version.

3.3.2 Detection of multi-polarity words

For detecting the multi-polarity words, we use the positive and negative labels of the training data. We make the assumption that positive words will mostly appear in positive tweets and negative words in negative tweets. Between two different corpora, we determine words with different polarity across corpora by using a χ^2 test on their profile of occurrence in positive and negative tweets in both corpora. The risk of false positive is set to 0.05. The words are also selected only if they occur more often than a given threshold. For the SemEval task B, we apply this detection for each domain. Each time, we detect the words that change their polarity between a specific domain and all the others. For example, the word "black" is detected as positive in the second domain, related to the election of Barack Obama, and neutral in the rest of the tweets. At the end of this procedure, we have 5 collections of words which change their polarity (one different collection for each domain). These collections are rather small: from 21 to 61 multi-polarity words are detected depending on the domain and the parameters.

3.3.3 Differentiation of multi-polarity words

We tested different strategies in order to integrate the domain information in the Sentiment Classification in Twitter task.

- Domain-specific: 5 different classifiers are trained on the domain specific subpart of the tweets, without change on the data.
- Diff-topic: 5 different classifiers are trained on the whole corpus, where the detected multi-polarity words are differentiated into "word-domainX" and "word-other".
- Change-all: only 1 classifier is trained. Similar to the previous one, except all the differentiations are made at the same time.
- Keep-topic: 5 different classifiers are trained. The detected multi-polarity words are kept inside their domain and deleted in the others.
- Remove-all: 5 different classifiers are trained. The detected multi-polarity words are deleted inside and outside their domain.

For the *change-all* version, we use only one classifier: all test tweets are classified using the same classifier. In the other versions, we obtain 5 classifiers. For each test tweet, we determine its domain profile using topic models of LDA. Then we use a mix of all the classifiers with weighting according to the LDA mixture[2]. The *domain-specific* version gives worse

[2]The weight is the exponential of the LDA score.

results than the baseline trained on the whole original corpus and is not represented on the figures.

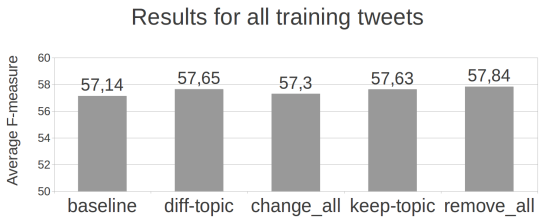

Figure 2: Average F-measure results for the best set of parameters for each method.

We tested all these versions with two training sets: first, using all the training tweets to train the classifiers (Figure 2) and secondly, only the tweets for which a domain can be confidently attributed (at least a 75% score from the LDA model) (Figure 3). In this case, the training set contains 2889 tweets. The run submitted to SemEval corresponds to the *change-all* version, trained with all the training tweets.

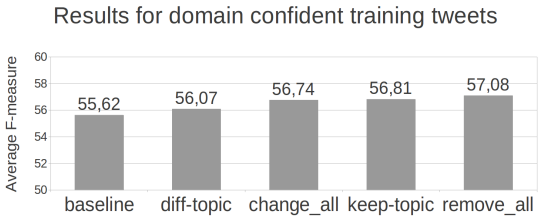

Figure 3: Average F-measure results for the best set of parameters for each method.

Empirically, we set the threshold for the number of occurrences to 10 in the first experiment and only to 5 in the domain confident experiment, due to the smallest size of the training corpora.

3.4 Analysis of the result and discussion

Using a boosting method with lemma trigrams and emoticons features is a good fully automatic baseline. We are in the mid range of results of all the participants (19th out of 48 submissions for the tweets and 26th out of 42 submissions for the SMS). We try to include domain information to improve the opinion classification. As we don't have a reference domain differentiation for the tweets, we separate them using the LDA method. The *domain-specific* version, which does not take into account the multi-polarity words, degrades the performances(-1.85% in the first experiment, -2.8% in the second). On the contrary, all our versions which use multi-polarity words, especially remove-all version, improve the F-measure. The final improvement is small but it has to be related to the small number of multi-polarity words we have detected (in average, 36 words per domain). We think that the tweet collection is too small for the χ_2 test to detect a lot of words with enough confidence. For comparison, in our experiment on reviews, we detected about 400 multi-polarity words per domain. It is also worth noticing that for the domain confident experiment, the improvement is more sensible (+1.46% versus +0.70%) even if the absolute value of the score is not better, due to a much smaller training data. It's a good argument for our method. Another question is about the method used to separate the tweets into different domains. We plan to have more control on the domains by using a more supervised method based on the categories of Wikipedia.

4 Conclusion

In this paper, we presented our contribution to SemEval 2013 task 2: Sentiment Analysis in Twitter. For the Contextual Polarity Disambiguation subtask, we described a very efficient and robust method based on a sentiment lexicon associated with a polarity shift detector and a tree based classification. As for the Message Polarity Classification, we focused on the impact of domain information. With only 4899 training tweets, we achieve good performances and we demonstrate that words with changing polarity can influence the classification performance.
One of the challenges of this SemEval task was to see how well sentiment analysis models trained using Twitter data would generalize to a SMS dataset. Looking at our result but also at the submissions of other participants, a drop of performance can be observed between the results on the Twitter and SMS test datasets. In (Hu et al., 2013), the authors perform a thorough study on the differences between the language used on Twitter and that of SMS messages and chat. They find that Twitter language is more conservative and less informal than SMS and online chat and that the language of Twitter can be seen as a projection of a formal register in a restricted space. This is a good indicator to the difficulty of using a Twitter centered system on a SMS dataset.

Acknowledgments

This work was partly supported by the MUCKE project (http://ifs.tuwien.ac.at/˜mucke/) through a grant from the French National Research Agency (ANR), FP7 CHIST-ERA Programme (ANR-12-CHRI-0007-04).

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th conference on International Language Resources and Evaluation (LREC10), Valletta, Malta, May*.

Romaric Besançon, Gaël de Chalendar, Olivier Ferret, Faiza Gara, Olivier Mesnard, Meriama Lab, and Nasredine Semmar. 2010. Lima : A multilingual framework for linguistic analysis and linguistic resources development and evaluation. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of LREC'10*, Valletta, Malta, may. European Language Resources Association (ELRA).

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Lu Chen, Wenbo Wang, Meenakshi Nagarajan, Shaojun Wang, and Amit P Sheth. 2012. Extracting diverse sentiment expressions with target-dependent polarity from twitter. *Proceedings of ICWSM*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 241–249. Association for Computational Linguistics.

Alexandru Lucian Ginsca. 2012. Fine-grained opinion mining as a relation classification problem. In *2012 Imperial College Computing Student Workshop*, volume 28, pages 56–61. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12.

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

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Yuheng Hu, Kartik Talamadupula, and Subbarao Kambhampati. 2013. Dude, srsly?: The surprisingly formal nature of twitters language. *Proceedings of ICWSM*.

Thorsten Joachims. 1999. Making large scale svm learning practical.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Morgane Marchand. 2013. Fouille dopinion: ces mots qui changent de polarité selon le domaine. In *Proceedings of the 8e Rencontres Jeunes Chercheurs en Recherche dInformation (RJCRI)*.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of EACL*, volume 6, pages 113–120.

R. Navigli. 2012. A quick tour of word sense disambiguation, induction and related approaches. *SOFSEM 2012: Theory and Practice of Computer Science*, pages 115–129.

Robert E Schapire and Yoram Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354. Association for Computational Linguistics.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*, 35.

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Alan Ritter, Sara Rosenthal, and Veselin Stoyanov. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Yasuhisa Yoshida, Tsutomu Hirao, Tomoharu Iwata, Masaaki Nagata, and Yuji Matsumoto. 2011. Transfer learning for multiple-domain sentiment analysis - identifying domain dependent/independent word polaritys. In *AAAI*.