BOUNCE: Sentiment Classification in Twitter using Rich Feature Sets

Nadin Kökciyan†, Arda Çelebi†, Arzucan Özgür, Suzan Üsküdarlı

Department of Computer Engineering
Bogazici University
Istanbul, Turkey
{nadin.kokciyan,arda.celebi,arzucan.ozgur,suzan.uskudarli}@boun.edu.tr

Abstract

The widespread use of Twitter makes it very interesting to determine the opinions and the sentiments expressed by its users. The shortness of the length and the highly informal nature of tweets render it very difficult to automatically detect such information. This paper reports the results to a challenge, set forth by SemEval-2013 Task 2, to determine the positive, neutral, or negative sentiments of tweets. Two systems are explained: System A for determining the sentiment of a phrase within a tweet and System B for determining the sentiment of a tweet. Both approaches rely on rich feature sets, which are explained in detail.

1 Introduction

Twitter consists of a massive number of posts on a wide range of subjects, making it very interesting to extract information and sentiments from them. For example, answering questions like 'What do Twitter users feel about the brand X?' are quite interesting. The constrained length and highly informal nature of tweets presents a serious challenge for the automated extraction of such sentiments.

Twitter supports special tokens (i.e. mentions and hashtags), which have been utilized to determine the sentiment of tweets. In (Go et al., 2009), emoticons are used to label tweets. In (Davidov et al., 2010), Twitter emoticons as well as hashtags are used to label tweets. O'Connor et al. (2010) demonstrated a correlation between sentiments identified in public opinion polls and those in tweets. A subjectivity lexicon was used to identify the positive and negative words in a tweet. In (Barbosa and Feng, 2010), subjective tweets are used for sentiment classification. They propose the use of word specific (e.g. POS tags) and tweet specific (e.g. presence of a link) features. Most of these studies use their own annotated data sets for evaluation, which makes it difficult to compare the performances of their proposed approaches.

Sentiment Analysis in Twitter 2013 (SemEval 2013 Task 2) (Wilson et al., 2013) presented a challenge for exploring different approaches examining sentiments conveyed in tweets: interval-level (phrase-level) sentiment classification (TaskA) and message-level sentiment classification (TaskB). Sentiment are considered as *positive, negative,* or *neutral*. For TaskA, the goal is to determine the sentiment of an interval (consecutive word sequence) within a tweet. For TaskB, the goal is to determine sentiment of an entire tweet. For example, let's consider a tweet like '*Can't wait* until the DLC for ME3 comes out tomorrow. *:-)*'. For TaskA, the interval 0-1 (*Can't wait*) is 'positive' and the interval 10-10 (*:-)*) is 'positive'. For TaskB, this tweet is 'positive'.

In this paper, we present two systems, one for TaskA and one for TaskB. In both cases machine learning methods were utilized with rich feature sets based on the characteristics of tweets. Our results suggest that our approach is promising for sentiment classification in Twitter.

2 Approach

The task of detecting the sentiments of a tweet or an interval therein, is treated as a classification of

† These authors contributed equally to this work

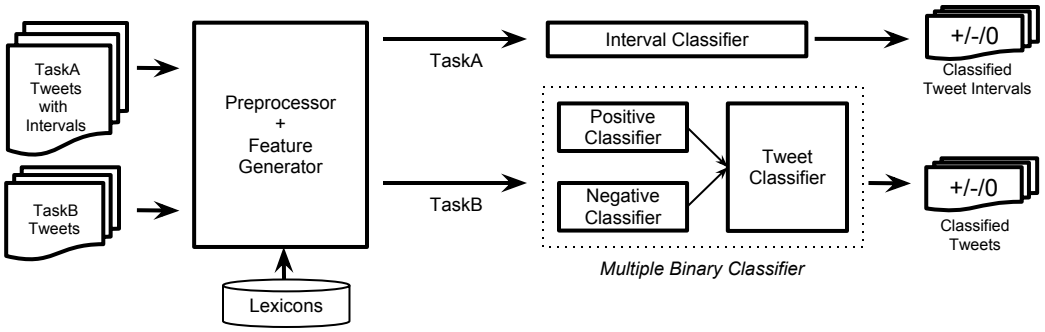

Figure 1: The Overview of BOUNCE System

tweets into positive, negative, or neutral sets. Figure 1 gives the overview of our approach. The Preprocessor module tokenizes the tweets that are used by the Feature Generator. At this stage, the tweets are represented as feature vectors. For TaskA, the feature vectors are used by the Interval Classifier that predicts the labels of the tweet intervals. For TaskB, the feature vectors are used by the Positive Classifier and the Negative Classifier which report on the positivity and negativity of the tweets. The Tweet Classifier determines the tweet labels using a rule-based method. Each step is described in detail in the following subsections.

2.1 Lexicons

The core of our approach to sentiment analysis relies on word lists that are used to determine the positive and negative words or phrases. Several acquired lists are used in addition to one that we curated. AFINN (Nielsen, 2011) is the main sentiment word list including 2477 words rated between -5 to 5 for valence. SentiWordNet (Baccianella et al., 2010), derived from the Princeton English WordNet (Miller, 1995), assigns positive, negative, or objective scores to each synset in WordNet. We considered the average of a word's synsets as its SentiWordNet score. Thus, synsets are disregarded and no disambiguation of the sense of a word in a given context is done. The SentiWordNet score of a word is not used if it has objective synsets, since it indicates that the word might have been used in an objective sense. We use a list of emotion words and categories that is created by DeRose[1]. Furthermore, a slang dictionary downloaded from the Urban Dictionary[2] containing over 16,000 phrases (with no sentiment) is used. Finally, we curated a sentiment word list initiated with a list of positive and negative words obtained from General Inquirer (Stone et al., 1966), and refined by sentiment emitting words from a frequency-based ordered word list generated from the training data set of SemEval-2013 Task A. Naturally, this list is more specialized to the Twitter domain.

2.2 Preprocessing

Prior to feature generation, tweets were preprocessed to yield text with more common wording. For this, CMU's Ark Tokenizer and Part-of-Speech (POS) Tagger (Gimpel et al., 2011), which has been specifically trained for tweets, was used. Tweets are tokenized and POS tagged.

2.3 Feature Sets

In addition to the lexical or syntactic characteristics, the manner in which tweets are written may reveal sentiment. Orthogonal shapes of words (esp. fully or partially capitalized words), expressions of a single word or a phrase in the form of a hashtag, positions of certain tokens in a tweet are prominent characteristics of tweets. In addition to these, tweets may convey multiple sentiments. This leads to sequence-based features, where we append features for each sentiment emitted by a word or a phrase in a tweet. Moreover, since TaskA asks for sentiment of intervals in a tweet, we also engineer features to catch clues from the surrounding context of the interval,

[1] http://derose.net/steve/resources/emotionwords/ewords.html

[2] http://www.urbandictionary.com

such as the sentiments and lengths of the neighboring intervals. For TaskB, the usage of hashtags and last words in tweets were occasionally sentimental, thus we considered them as features as well. We explain all features in detail in Section 3.

2.4 Classification

Maximum entropy models (Berger et al., 1996) have been used in sentiment analysis (Fei et al., 2010). They model all given data and treat the remainder as uniform as possible making no assumptions about what is not provided. For this, TaskA system uses the MaxEnt tool (Zhang, 2011).

Naive Bayes is a simple probabilistic model based on *Bayes' Theorem* that assumes independence between features. It has performed well in sentiment classification of Twitter data (Go et al., 2009; Bifet and Frank, 2010). TaskB data was not evenly distributed. There were very few negative tweets compared to positive tweets. Using a single classifier to distinguish the classes from each other resulted in poor performance in identifying negative tweets. Therefore, TaskB system utilizes multiple binary classifiers that use the one-vs-all strategy. Maximum Entropy and Naive Bayes models were considered and the model that performed best on the development set was chosen for each classifier. As a result, the positive classifier (B_{pos}) is based on the Maximum Entropy model, whereas the negative classifier (B_{neg}) is based on Naive Bayes. TaskB system uses the Natural Language Toolkit (Loper and Bird, 2002).

3 Systems

In this section, TaskA and TaskB systems are explained in detail. All features used in the final experiments for both tasks are shown in Table 1.

3.1 TaskA System

TaskA is a classification task where we classify a given interval as having positive, negative or neutral sentiment. TaskA feature sets are shown in Table 1.

lexical features: These features use directly words (or tokens) from tweets as features. *single-word* feature uses the word of the single-word intervals, whereas *slang* features are created for matching uni-grams and bi-grams from our slang dictionary. We also use emoticons as features, as well as the words or phrases that emit emotion according to the lexicons described in Section 2.1.

score-based features: These features use the scores obtained from the AFINN and SentiWordNet (SWN) lexicons. We use separate scores for the positive and negative sentiments, since one interval may contain multiple words with opposite sentiment. In case of multiple positive or negative occurances, we take the arithmetic mean of those.

shape-based features: These features capture the length of an interval, whether it contains a capitalized word or all words are capitalized, whether it contains a URL, or ends with an exclamation mark.

tag-based features: In addition to numeric values of sentiments, we use the tokens 'positive' and 'negative' to express the type of sentiment. When multiple words emit a sentiment in a given interval, their corresponding tokens are appended to create a single feature out of it, *sequences*. Moreover, we have another set of features which also contains the POS tags of these sentiment words.

indicator features: These features are used in order to expose how many sentiment emitting words from our currated large lexicon exist in a given interval. *hasNegation* indicates the presence of a negation word like *not* or *can't* in the interval, whereas *numOfPosIndicators* and *numOfNegIndicators* gives the number of tokens that convey positive and negative sentiment, respectively.

context features: In addition to the features generated from the given interval, these features capture the context information from the neighboring intervals. Feature *surroundings* combines the length of the interval along with the lengths of the intervals on both sides, whereas *surrounding-shape* and *extra-surrounding-shape* features use number of positive and negative sentiment indicators for the intervals. We also use their normalized forms (those starting with *norm-*) where we divide the number of indicators by the length of the interval. Features with *-extra-* use two adjacent intervals from both sides. Intervals that are not available are represented with *NA*.

3.2 TaskB System

TaskB is a classification task where we determine the sentiment (positive, negative, or neutral) of a tweet. TaskB system uses a rule-based method to

Feature Set	Feature	Example Feature Instance	used by
lexical-based	single-word-*	*single-word-worst*	A, B
	slang-*	*slang-shit*	A, B_{pos}
	emoticons-*	*emoticons-:)*	A
	emitted-emotions-*	*emitted-emotions-angry*	A, B
score-based	afinn-positive:#, afinn-negative:#	*afinn-positive:4, afinn-negative:-2*	A, B
	swn-positive:#, swn-negative:#	*swn-positive:2, swn-negative:-3*	A
shape-based	length-#	*length-10*	A
	hasAllCap-T/F	*hasAllCap-T*	A
	fullCap-T/F	*fullCap-T*	A
	hasURL-T/F	*hasURL-F*	A, B
	endsWExlamation-T/F	*endsWExlamation-T*	A, B_{neg}
tag-based	our-seq-*	*our-seq-positive-positive*	A, B
	our-tag-seq-*, swn-seq-*, swn-tag-seq-*	*afinn-seq-positive-a-positive-n*	A
	afinn-seq-*, afinn-tag-seq-*	*afinn-seq-positive-a-negative-n*	A
indicators	hasNegation-T/F	*hasNegation-F*	A
	numOfPosIndicators-#	*numOfPosIndicators-2*	A
	numOfNegIndicators-#	*numOfNegIndicators-0*	A
context	surroundings-#-#-#	*surroundings-1-2-NA*	A
	surr-shape-#-#-#	*surrounding-shape-NA-2-1*	A
	extra-surr-shape-#-#-#-#-#	*extra-surr-shape-NA-2-1-0-1*	A
	norm-surr-shape-#-#-#	*norm-surr-shape-0.5-0.2-0.0*	A
	norm-extra-surr-shape-#-#-#-#-#	*norm-extra-surr-shape-NA-0.5-0.2-0.0-0.2*	A
	left-sentiment-*, right-sentiment-*	*left-sentiment-positive*	A
twitter-tags	hasEmoticon-T/F	*hasEmoticon-T*	B
	hasMention-T/F	*hasMention-T*	B
	hasHashtag-T/F	*hasHashtag-F*	B
	[emoticon|mention|hash]-count-#	*mention-count-3*	B
repetition	unigram-*$_n$	*unigram-[no+]*	B
	$character-count-#	*o-count-7*	B
lastword	lastword-*$_n$	*lastword-[OMG+]*	B
	lastwordshape-*	*lastwordshape-XXXX*	B
chat	chatword-*	for word 'gz': *chatword-congratulations*	B
interjection	interjection-*$_n$	*interjection-[lo+l]*	B
negation	negword-*$_n$	*negword-never*	B_{neg}
	negword-count-#	*negword-count-3*	B_{neg}
	negcapword-count-#	*negcapword-count-1*	B_{neg}
hash	hashword-*	*hashword-good*	B
	hashtag-#*	*hashtag-#good*	B
	hash-sentiment-[positive|negative]	*hash-sentiment-positive*	B
lingemotion	[noun|verb|adverb|adjective]-*$emotion*	*noun-fear*	B
oursent	oursent-*	for tweet: a $nice$ morning.. I $hate$ work.. $damn$! *oursent-nice*, *oursent-hate*, *oursent-damn*	B
	oursent-longseq-*	*oursent-longseq-pnn*	B
	oursent-shortseq-*	*oursent-shortseq-pn*	B
	oursent-first-last-*	*oursent-first-last-pn*	B
afinn-phrases	phrase-firstsense-[positive|negative]	*phrase-firstsense-positive*	B
	phrase-lastsense-[positive|negative]	*phrase-lastsense-negative*	B
	afinnword-*	*afinnword-nice*, *afinnword-hate*, *afinnword-damn*	B
	afinn-firstsense-[positive|negative]	*afinn-firstsense-positive*	B
	afinn-lastsense-[positive|negative]	*afinn-lastsense-positive*	B
emo	emo-pattern-*	for =) : *emo-pattern-HAPPY*	B

Table 1: Feature sets used in TaskA and TaskB

Dataset	Type	Positive	Negative	Neutral+Objective	Tot. No. of Instances
TaskA	Training	5290 (5865)	2771(3120)	16118 (17943)	24179 (26928)
	Development	589 (648)	392 (430)	1993 (2202)	2974 (3280)
	Test	2734	1541	160	4435
TaskB	Training	3274 (3640)	1291 (1458)	4155 (4586)	8720 (9684)
	Development	523 (575)	309 (340)	674 (739)	1506 (1654)
	Test	1572	601	1640	3813

Table 2: Number of instances used in TaskA and TaskB

decide on the sentiment label of a tweet. For each tweet, the probabilities of belonging to the positive class ($Prob_{pos}$) and negative class ($Prob_{neg}$) are computed by the B_{pos} and B_{neg} classifiers, respectively. If $Prob_{pos}$ is greater than $Prob_{neg}$, and greater than a predefined threshold, then the tweet is classified as 'positive', otherwise it is classified as 'neutral'. On the other hand, if $Prob_{neg}$ is greater than $Prob_{pos}$, and greater than the predefined threshold, then the tweet is classified as 'negative', otherwise it is classified as 'neutral'. The threshold is set to 0.45, since it gives the optimal F-score on the development set. TaskB features along with examples are shown in Table 1.

twitter-tags: *hasEmoticon*, *hasMention*, *hasURL*, and *hasHashtag* indicate whether the corresponding term (e.g. mention) exists in the tweet.

repetition: Words with repeating letters are added as a feature $*_n$. $*_n$ represents the normalized version (i.e., no repeating letters) of a word. For example, 'nooooooo' is shortened to *[no+]*. We also keep the count of the repeated character.

wordshape: Shape of each word in a tweet is considered. For example, the shape of 'NOoOo!!' is 'XXxXx!!'.

lastword: The normalized form and the shape of the last word are used as features. For example, if the lastword is 'OMGG', then *lastword* '[OMG+]' and *lastwordshape* 'XXXX' are used as features.

chat: A list of chat abbreviations that express sentiment is manually created. Each abbreviation is replaced by its corresponding word.

interjection: An interjection is a word that expresses an emotion or sentiment (e.g. hurraah, loool). Interjection word$_n$ is used as a feature.

negation: We manually created a negation list extended by word clusters from (Owoputi et al., 2013). A negation word is represented by spellings such as not, n0t, and naht. Each negation word$_n$ (e.g neve[r+]) is considered. We keep the count of negation words and all capitalized negation words.

hash: If the hashtag is '#good' then *#good* and *good* become hash features. If the hashtag is a sentiment expressing word according to our sentiment word list, then we keep the sentiment information.

lingemotion: Nodebox Linguistics[3] package gives emotional values of words for expressions of emotions such as fear and sadness. POS augmented expression information is used as a feature.

oursent: Each word in a tweet that exists in our sentiment word list is considered. When multiple sentiment expressing words are found, a sentiment sequence feature is used. *oursent-longseq* keeps the long sequence, whereas *oursent-shortseq* keeps same sequence without repetitive sentiments. We also consider the first and last sentiments emitted by a tweet.

afinn: We consider each word that exists in AFINN. If a negation exists before this word, the opposite sentiment is considered. For example, if a tweet contains the bigram 'not good', then the sentiment of the bigram is set to 'negative'. The AFINN scores of the positive and negative words, as well as the first and last sentiments emitted by the tweet are considered.

phrases: Each n-gram ($n > 1$) of a tweet that exists in our sentiment phrase list is considered.

afinn-phrases: Phrases are retrieved using the *phrases* feature. Each sentiment that appears in a phrase is kept, hence we obtain a sentiment sequence. The first and last sentiments of this sequence are also considered. Then, the phrases are removed from the tweet text and the *afinn* feature is applied.

emo: We manually created an emoticon list where

[3] http://nodebox.net/code/index.php/Linguistics

each term is associated with an emotion pattern such as HAPPY. These emotion patterns are used as a feature.

others: B_{pos} uses the *slang* feature from the lexical feature set, and B_{neg} uses *endsWExlamation* feature from the indicators feature set.

4 Experiments and Results

4.1 Data

The data set provided by the task organizers was annotated by using Amazon Mechanical Turk[4]. The annotations of the tweets in the training and development sets were provided to the task participants. However, the tweets had to be downloaded from Twitter by using the script made available by the organizers. We were unable to download all the tweets in the training and development sets, since some tweets were deleted and others were not publicly accessible due to their updated authorization status. The number of actual tweets (numbers in parentheses) and the number of collected tweets are shown in Table 2. Almost 10% of the data for both tasks are missing. For the test data, however, the tweets were directly provided to the participants.

4.2 Results on TaskA

We start our experiments with features generated from lexicons and emoticons. Called our baseline, it achieved an f-score of 47.8 on the devset in Table 3. As we add other features at each step, we reach an average f-score of 81.6 on the devset at the end. Among those features, the most contributing ones are lexical feature $single\text{-}word$, indicator feature $hasNegation$, and especially shape feature $length$. The success of the $length$ feature is mostly due to the nature of intervals, where the long ones tend to be neutral, and the rest are mostly positive or negative. Another noteworthy result is that our curated word list contributed more compared to the others. When the final model is used on the test set, we get the results in Table 5. Having low neutral f-score might be due to the fact that there were only a few neutral intervals in the test set, which might indicate that their characteristics may not be the same as the ones in the devset.

[4]https://www.mturk.com/mturk/

Added Features	**Avg. F-Score**
afinn-positive, afinn-negetive swn-positive, swn-negative, emoticons, emitted-emotions	47.8
+ hasAllCap, fullCap, hasURL, endsWExclamation	50.1
+ slang	51.5
+ single-word	56.8
+ afinn-seq, swn-seq, afinn-tag-seq, swn-tag-seq	57.7
+ our-seq, our-tag-seq	60.2
+ hasNegation	64.8
+ numOfPosIndicators, numOfNegIndicators	65.3
+ length	75.2
+ left-sentiment, right-sentiment	76.5
+ surroundings, surrounding-shape	78.9
+ extra-surrounding-shape	80.6
+ norm-surrounding-shape, norm-extra-surrounding-shape	**81.6**

Table 3: Macro-averaged F-Score on the TaskA dev. set

Added Features	**Average F-Score**
oursent (baseline)	58.59
+ afinn-phrases	64.64
+ tags + hash	65.43
+ interjection + chat	65.53
+ emo + lingemotion	65.92
+ repetition + lastword	66.01
+ negation + others	**66.32**

Table 4: Macro-averaged F-Score on the TaskB dev. set

4.3 Results on TaskB

The baseline model is considered to include *oursent* feature that gives an average f-score of 58.59. Next, we added the *afinn-phrases* feature which increased the average f-score to 64.64. This increase can be explained by the sentiment scores and sequence patterns that *afinn-phrases* is based on. Following that model, the other added features slightly increased the average f-score to 66.32 as shown in Table 4. The final model is used over the test set of TaskB, where we obtained an f-score of 63.53 as shown in Table 5.

	Class	Precision	Recall	F-Score
TestA	positive	89.7	88.3	89.0
	negative	86.6	82.7	84.6
	neutral	10.7	18.1	13.4
average(pos+neg)		88.15	85.5	**86.8**
TestB	positive	82.3	55.6	66.4
	negative	48.7	80.2	60.6
	neutral	68.2	73.3	70.7
average(pos+neg)		65.56	67.93	**63.53**

Table 5: Results on the test sets for both tasks

5 Conclusion

We presented two systems one for TaskA (a Maximum Entropy model) and one for TaskB (Maximum Entropy + Naive Bayes models) based on using rich feature sets. For Task A, we started with a baseline system that just uses ordinary features like sentiment scores of words. As we added new features, we observed that lexical features and shape-based features are the ones that contribute most to the performance of the system. Including the context features and the indicator feature for negations led to considerable improvement in performance as well. For TaskB, we first created a baseline model that uses sentiment words and phrases from the AFINN lexicon as features. Each feature that we added to the system resulted in improvement in performance. The *negation* and *endsWExclamation* features only improved the performance of the negative classifier, whereas the *slang* feature only improved the performance of the positive classifier.

Our results show that using rich feature sets with machine learning algorithms is a promising approach for sentiment classification in Twitter. Our TaskA system ranked 3rd among 23 systems and TaskB system ranked 4th among 35 systems participating in SemEval 2013 Task 2.

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 Seventh Conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 36–44, Stroudsburg, PA, USA. Association for Computational Linguistics.

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71.

Albert Bifet and Eibe Frank. 2010. Sentiment knowledge discovery in twitter streaming data. In *Proceedings of the 13th international conference on Discovery science*, DS'10, pages 1–15, Berlin, Heidelberg. Springer-Verlag.

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*, COLING '10, pages 241–249, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiaoxu Fei, Huizhen Wang, and Jingbo Zhu. 2010. Sentiment word identification using the maximum entropy model. In *International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE)*, pages 1–4.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 42–47. Association for Computational Linguistics.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Technical report, Stanford University.

Edward Loper and Steven Bird. 2002. Nltk: the natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.

Finn Å. Nielsen. 2011. A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*.

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010. From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL*.

Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. 1966. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press.

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Sara Rosenthal, Veselin Stoyanov, and Alan Ritter. 2013. SemEval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '13, June.

Le Zhang. 2011. Maximum entropy modeling toolkit for python and c++. `http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html`. Accessed: 2013-04-13.