Swiss-Chocolate: Sentiment Detection using Sparse SVMs and Part-Of-Speech n-Grams

Martin Jaggi
ETH Zurich
Zürich, Switzerland
jaggi@inf.ethz.ch

Fatih Uzdilli and **Mark Cieliebak**
Zurich University of Applied Sciences
Winterthur, Switzerland
{ uzdi, ciel } @zhaw.ch

Abstract

We describe a classifier to predict the message-level sentiment of English microblog messages from Twitter. This paper describes the classifier submitted to the SemEval-2014 competition (Task 9B). Our approach was to build up on the system of the last year's winning approach by NRC Canada 2013 (Mohammad et al., 2013), with some modifications and additions of features, and additional sentiment lexicons. Furthermore, we used a sparse (ℓ_1-regularized) SVM, instead of the more commonly used ℓ_2-regularization, resulting in a very sparse linear classifier.

1 Introduction

With the immense growth of user generated text online, the interest in automatic sentiment analysis of text has greatly increased recently in both academia and industry.

In this paper, we describe our approach for a modified SVM based classifier for short text as in Twitter messages. Our system has participated in the SemEval-2014 Task 9 competition, "Sentiment Analysis in Twitter, Subtask–B Message Polarity Classification" (Rosenthal et al., 2014). The goal is to classify a tweet (on the full message level) into the three classes positive, negative, and neutral. An almost identical competition was already run in 2013.

Our Results in the Competition. Our approach was ranked on the 8th place out of the 50 participating submissions, with an F1-score of 67.54 on the Twitter-2014 test set. The 2014 winning team obtained an average F1-score of 70.96.

(The more detailed rankings of our approach were 4th rank on the LiveJournal data, 5th on the SMS data (2013), 18th on Twitter-2013, and 16th on Twitter Sarcasm, see (Rosenthal et al., 2014) for full details and all results).

Data. In the competition, the tweets for training and development were only provided as tweet IDs. A fraction (10-15%) of the tweets were no longer available on twitter, which makes the results of the competition not fully comparable. For testing, in addition to last years data (tweets and SMS), new tweets and data from a surprise domain were provided. An overview of the data, which we were able to download, is shown in Table 1.

Table 1: Overview of the data we found available for training, development and testing.

Dataset	Total	Positive	Negative	Neutral
Train (Tweets)	8224	3058	1210	3956
Dev (Tweets)	1417	494	286	637
Test: Twitter2014	1853	982	202	669
Test: Twitter2013	3813	1572	601	1640
Test: SMS2013	2093	492	394	1207
Test: Tw2014Sarcasm	86	33	40	13
Test: LiveJournal2014	1142	427	304	411

2 Description of Our Approach

Compared to the previous NRC Canada 2013 approach (Mohammad et al., 2013), our main changes are the following three: First we use sparse linear classifiers instead of classical dense ones. Secondly, we drop n-gram features completely, in favor of what we call *part-of-speech n-grams*, which are n-grams where up to two tokens are the original ones, and the rest of the tokens is replaced by their corresponding POS tag (noun, verb, punctuation etc). Third, we added two new sentiment lexicons, containing numerical scores associated for all 3 classes (positive, neutral, negative), instead of just 2 as in classical po-

larity lexicons. All changes are described in more detail in Sections 4 and 3 below.

Performance. We tried to reproduce the same classifier as in (Mohammad et al., 2013) as a baseline for comparison.

Trying to quantify our contributions, when adding all our additional features and tricks described below, the score of our method increases from the baseline of 63.25 to 64.81 (on the Twitter-2013 test set), which is a gain of 1.56 points in F1.

Baseline Approach by NRC Canada 2013. Unfortunately our replica system of Mohammad et al. (2013) only achieved an F1-score of 63.25 on the Twitter-2013 test set, while their score in the 2013 competition on the same test set was 69.02, nearly 6 points higher in F1.

Part of this big difference might be explained by the fact that the exact same training sets are not available anymore. Other possibly more important differences are the SVM classifier variant used and class weighting (described in Section 4). Furthermore, we didn't implement all features in the exactly same way, see the more detailed description in Section 3.1.2 below. Although we had the impression that these changes individually had only a relatively minor effect, it might be that the changes together with the different training set add up to the difference in score.

3 Features

Before we describe the linear classifier in Section 4, we detail the used features for each tweet message. On average, we generated 843 features per tweet. For comparison, the average in our NRC Canada 2013 replica system was only 285. Most of the increase in features comes from the fact that we allowed for slightly longer n-grams (6 instead of 4), and substrings (length 6 instead of 5).

3.1 New Features

3.1.1 Part of Speech n-grams

We used the ArkTweetNLP structured prediction POS tagger provided by Owoputi et al. (2013) together with their provided standard model (model.20120919) suitable for twitter data.

Part of speech n-grams are n-grams where up to two tokens are kept as the original ones, and all other tokens are replaced by their corresponding POS tag (noun, verb, punctuation etc). We generated these modified n-grams for all possible positions of the one or two original tokens within the n positions, for $3 \leq n \leq 6$.

As features for a classifier, we found POS n-grams at least as useful (with some more robustness) as the n-grams themselves. In our final approach, we dropped the use of n-grams completely, and only used POS n-grams instead. The idea of replacing some of the tokens by their POS tag is also investigated by Joshi and Penstein-Rosé (2009), where the authors used $n \leq 3$.

3.1.2 Various Changes Compared to NRC Canada 2013

- We do not allow n-grams (or POS n-grams) to span over sentence boundaries.
- Substrings of length up to 6 (instead of 5).
- Substring features are weighted increasingly by their length (weights $0.7 \cdot \{1.0, 1.1, 1.2, 1.4, 1.6, 1.9\}$ for lengths $3, 4, \dots$)
- Instead of the score itself, we used the sigmoid value $s(t) = 1/(1 + e^{-t}))$ of each lexicon score. For each lexicon, the 4 scores were the same as in (Mohammad et al., 2013), i.e. per tweet, we use the number of tokens appearing in the lexicon, the sum and the max of the scores, and the last non-zero score.

We skipped some features from the baseline approach (because their effect was not significant in our setting): Elongated words (number of words with one character repeated more than two times), and word clustering. Also, we had a slightly simplified variant of how to use the lexicon scores. We didn't count the lexicon scores separately per emotion (pos and neg), but only altogether.

3.2 Existing Features

Text Preprocessing. A good tokenization seems very important for twitter data. We used the popular tokenizer ArkTweetNLP (Owoputi et al., 2013) which is suitable for tweets. All text was transformed to lowercase (except for those features in (Mohammad et al., 2013) which use case information). As usual, URLs were normalized to http://someurl and twitter user IDs to @someuser.

We also employed the usual marking of negated contexts of a sentence as in (Pang et al., 2002),

using the list of negation words from Christopher Potts' sentiment tutorial[1].

4 Classifier

We used a linear support vector machine (SVM) classifier, which is standard for text data. The LibLinear package (Fan et al., 2008) was employed for training the multi-class classifier.

Multi-Class Formulation, and Class Weights. We found significant performance changes depending on which type of multi-class SVM, and also which regularizer (ℓ_1- or ℓ_2-norm) is used. For the multi-class variant, we found the *one-against-all* models to perform slightly better than the Crammer and Singer (2001) formulation.

More importantly, since the 3 classes (positive, negative and neutral) are quite unbalanced in size in the training set, it is crucial to set a good weight for each class in the SVM. We used $(4.52, 1.38, 1.80)$, which corresponds to the twice the ratio of each class compared to the average class size.

Sparse Linear Classifiers. In our setting, an ℓ_1-regularized squared loss SVM (one-against-all) performed best (this is mode L1R_L2LOSS_SVC in LibLinear), despite the fact that ℓ_2-regularization is generally more commonly used in text applications. We used $C = 0.055$ for the regularization parameter, and $\varepsilon = 0.003$ as the optimization stopping criterion. We did not employ any kernel, but always used linear classifiers.

Another benefit of the ℓ_1-regularization is that the resulting classifier is extremely sparse and compact, which significantly accelerates the evaluation of the classifier on large amounts of text, e.g. for testing. Our final classifier only uses 1985 non-zero features (1427 unigram/substrings, and 558 other features, such as lexicon scores, n-grams, POS n-grams, as explained in the previous Section 3).

As the resulting classifier is so small, it is also relatively easy to read and interpret. We have made our final classifier weights publicly available for download as a text file[2] . Every line contains the feature description followed by the 3 weights corresponding to the 3 sentiment classes.

Our final classifier was trained on 9641 tweets, which are all we could download from the IDs given in this years train and dev set.

5 Lexicons

A sentiment lexicon is a mapping from words (or n-grams) to an association score corresponding to positive or negative sentiment. Such lists can be constructed either from manually labeled data (supervised), or automatically labeled data (unsupervised) as for example tweets with a positive or negative smiley. We used the same set of lexicons as in (Mohammad et al., 2013), with one addition:

5.1 A Lexicon for 3-Class Classification

Our main new addition was another type of lexicon, which not only provides one score per word, but 3 of them, (being the association to positive, negative and neutral). The idea here is to improve on the discrimination quality, especially for neutral text, and treat all 3 labels in this multi-class task the same way, instead of just 2 as in the previous approaches.

Data. We found it challenging to find good datasets to build such a lexicon. We again used the Sentiment140 corpus (Go et al., 2009) (containing tweets with positive or negative emoticons). Using a subset of 100k positive and 100k negative ones, we added a set of 100k arbitrary (hopefully neutral) tweets. The neutral set was chosen randomly from the thinknook.com dataset[3] of 1.5mio tweets (from which we ignored the provided labels, and counted the tweets as neutral).

We did the same with the movie reviews from the recent kaggle competition on annotated reviews from the rotten-tomatoes website[4]. We automatically built a lexicon from 100k texts in this dataset, with the data balanced equally for the three classes.

Features Used in the Lexicon. To construct the lexicon, we extracted the POS n-grams (as we described in Section 3.1.1 above) from all texts. In comparison, Mohammad et al. (2013) used non-contiguous n-grams (unigram–unigram, unigram–bigram, and bigram–bigram pairs). We only used POS n-grams with 2 tokens kept original, and the

[1] http://sentiment.christopherpotts.net/lingstruc.html

[2] http://www.m8j.net/sentiment/

[3] http://thinknook.com/twitter-sentiment-analysis-training-corpus-dataset-2012-09-22/

[4] http://www.kaggle.com/c/sentiment-analysis-on-movie-reviews/data

remaining ones replaced by their POS tag, with n ranging from 3 to 6.

Building the Lexicon. While in (Mohammad et al., 2013), the score for each n-gram was computed using point-wise mutual information (PMI) with the labels, we trained a linear classifier on the same labels instead. The lexicon weights are set as the resulting classifier weights for our (POS) n-grams. We used the same type of sparse SVM trained with LibLinear, for 3 classes, as in the final classifier.

Download of the Lexicons. We built 4 lexicons as described above. Thanks to the sparsity of the linear weights from the SVM, they are again relatively small, analogous to the final classifier. We also provide the lexicons for download as text files[5].

5.2 Existing Lexicons

Lexicons from Manually Labeled Data. We used the same 3 existing sentiment lexicons as in (Mohammad et al., 2013). All lexicons give a single score for each word (if present in the lexicon). Those existing lexicons are: NRC Emotion Lexicon (about 14k words), the MPQA Lexicon (about 8k words), and the Bing Liu Lexicon (about 7k words).

Lexicons from Automatically Labeled Data. The NRC hashtag sentiment lexicon was generated automatically from a set of 775k tweets containing a hashtag of a small predefined list of positive and negative hashtags (Mohammad et al., 2013). Lexicon scores were trained via PMI (point-wise mutual information). Scores are not only available for words, but also unigram–unigram, unigram–bigram, and bigram–bigram pairs (that can be non-contiguous in the text).

The Sentiment140 lexicon (Go et al., 2009) was generated automatically from a set of 1.6 million tweets containing a positive or negative emoticon. This uses the same features and scoring as above.

6 Conclusion

We have described an SVM classifier to detect the sentiment of short texts such as tweets. Our system is built up on the approach of NRC Canada (Mohammad et al., 2013), with several modifications and extensions (e.g. sparse linear classifiers, POS-n-grams, new lexicons). We have seen that our system significantly improves the baseline approach, achieving a gain of 1.56 points in F1 score.

We participated in the SemEval-2014 competition for Twitter polarity classification, and our system was among the top ten out of 50 submissions, with an F1-score of 67.54 on tweets.

For future work, it would be interesting to incorporate our improvements into the most recent version of NRC Canada or similar systems, to see how much one could gain there.

[5] http://www.m8j.net/sentiment/

References

Crammer, K. and Singer, Y. (2001). On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *JMLR*, 2:265–292.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A Library for Large Linear Classification. *JMLR*, 9:1871–1874.

Go, A., Bhayani, R., and Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. Technical report, The Stanford Natural Language Processing Group.

Joshi, M. and Penstein-Rosé, C. (2009). Generalizing dependency features for opinion mining. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, p 313–316, Singapore. Association for Computational Linguistics.

Mohammad, S. M., Kiritchenko, S., and Zhu, X. (2013). NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *SemEval-2013 - Proceedings of the International Workshop on Semantic Evaluation*, pages 321–327, Atlanta, Georgia, USA.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013). Improved Part-Of-Speech Tagging for Online Conversational Text with Word Clusters. In *Proceedings of NAACL-HLT*, pages 380–390.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. In *ACL-02 conference*, pages 79–86, Morristown, NJ, USA. Association for Computational Linguistics.

Rosenthal, S., Ritter, A., Nakov, P., and Stoyanov, V. (2014). SemEval-2014 Task 9: Sentiment Analysis in Twitter. In *SemEval 2014 - Proceedings of the Eighth International Workshop on Semantic Evaluation*, Dublin, Ireland.