SZTE-NLP: Sentiment Detection on Twitter Messages

Viktor Hangya, Gábor Berend, Richárd Farkas
University of Szeged
Department of Informatics
hangyav@gmail.com, {berendg,rfarkas}@inf.u-szeged.hu

Abstract

In this paper we introduce our contribution to the SemEval-2013 Task 2 on "Sentiment Analysis in Twitter". We participated in "task B", where the objective was to build models which classify tweets into three classes (positive, negative or neutral) by their contents. To solve this problem we basically followed the supervised learning approach and proposed several domain (i.e. microblog) specific improvements including text preprocessing and feature engineering. Beyond the supervised setting we also introduce some early results employing a huge, automatically annotated tweet dataset.

1 Introduction

In the past few years, the popularity of social media has increased. Many studies have been made in the area (Jansen et al., 2009; O'Connor et al., 2010; Bifet and Frank, 2010; Sang and Bos, 2012). People post messages on a variety of topics, for example products, political issues, etc. Thus a big amount of user generated data is created day-by-day. The manual processing of this data is impossible, therefore automatic procedures are needed.

In this paper we introduce an approach which is able to assign sentiment labels to Twitter messages. More precisely, it classifies tweets into positive, negative or neutral polarity classes. The system participated in the *SemEval-2013 Task 2: Sentiment Analysis in Twitter, Task–B Message Polarity Classification* (Wilson et al., 2013). In our approach we used a unigram based supervised model because it has been shown that it works well on short messages like tweets (Jiang et al., 2011; Barbosa and Feng, 2010; Agarwal et al., 2011; Liu, 2010). We reduced the size of the dictionary by normalizing the messages and by stop word filtering. We also explored novel features which gave us information on the polarity of a tweet, for example we made use of the acronyms in messages.

In the "constrained" track of *Task–B* we used the given training and development data only. For the "unconstrained" track we downloaded tweets using the Twitter Streaming API[1] and automatically annotated them. We present some preliminary results on exploiting this huge dataset for training our classifier.

2 Approach

At the beginning of our experiments we used a unigram-based supervised model. Later on, we realized that the size of our dictionary is huge, so in the normalization phase we tried to reduce the number of words in it. We investigated novel features which contain information on the polarity of the messages. Using these features we were able to improve the precision of our classifier. For implementation we used the MALLET toolkit, which is a Java-based package for natural language processing (McCallum, 2002).

2.1 Normalization

One reason for the unusually big dictionary size is that it contains one word in many forms, for exam-

[1] https://dev.twitter.com/docs/streaming-apis/streams/public

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 549–553, Atlanta, Georgia, June 14-15, 2013.

ple in upper and lower case, in a misspelled form, with character repetition, etc. On the other hand, it contained numerous special annotations which are typical for blogging, such as Twitter-specific annotations, URL's, smileys, etc. Keeping these in mind we made the following normalization steps:

- First, in order to get rid of the multiple forms of a single word we converted them into lower case form then we stemmed them. For this purpose we used the Porter Stemming Algorithm.
- We replaced the @ and # Twitter-specific tags with the *[USER]* and *[TAG]* notations, respectively. Besides we converted every URL in the messages to the *[URL]* notation.
- Smileys in messages play an important role in polarity classification. For this reason we grouped them into positive and negative smiley classes. We considered *:), :-),:), :D, =), ;), ;), (:* and *:(, :-(, : (,):,) :* smileys as positive and negative, respectively.
- Since numbers do not contain information regarding a message polarity, we converted them as well to the *[NUMBER]* form. In addition, we replaced the question and exclamation marks with the *[QUESTION_MARK]* and *[EXCLAMATION_MARK]* notations. After this we removed the unnecessary characters `'"#$%&()*+,./:;<=>\^{}~`, with the exception that we removed the `'` character only if a word started or ended with it.
- In the case of words which contained character repetitions – more precisely those which contained the same character at least three times in a row –, we reduced the length of this sequence to three. For instance, in the case of the word *yeeeeahhhhhhh* we got the form *yeeeahhh*. This way we unified these character repetitions, but we did not loose this extra information.
- Finally we made a stop word filtering in order to get rid of the undesired words. To identify these words we did not use a stop word dictionary, rather we filtered out those words which appeared too frequently in the training corpus. We have chosen this method because we would like to automatically detect those words which are not relevant in the classification.

Before the normalization step, the dictionary contained approximately $41,000$ words. After the above introduced steps we managed to reduce the size of the dictionary to $15,000$ words.

2.2 Features

After normalizing Twitter messages, we searched for special features which characterize the polarity of the tweets. One such feature is the polarity of each word in a message. To determine the polarity of a word, we used the SentiWordNet sentiment lexicon (Baccianella et al., 2010). In this lexicon, a positive, negative and an objective real value belong to each word, which describes the polarity of the given word. We consider a word as positive if the related positive value is greater than 0.3, we consider it as negative if the related negative value is greater than 0.2 and we consider it as objective if the related objective value is greater than 0.8. The threshold of the objective value is high because most words are objective in this lexicon. After calculating the polarity of each word we created three new features for each tweet which are the number of positive, negative and objective words, respectively. We also checked if a negation word precedes a positive or negative word and if so we inverted its polarity.

We also tried to group acronyms by their polarity. For this purpose we used an acronym lexicon which can be found on the `www.internetslang.com` website. For each acronym we used the polarity of each word in the acronym's description and we determined the polarity of the acronym by calculating the rate of positive and negative words in the description. This way we created two new features which are the number of positive and negative acronyms in a given message.

Our intuition was that people like to use character repetitions in their words for expressing their happiness or sadness. Besides normalizing these tokens (see Section 2.1), we created a new feature as well, which represents the number of this kind of words in a tweet.

Beyond character repetitions people like to write words or a part of the text in upper case in order to

call the reader's attention. Because of this we created another feature which is the number of upper case words in the given text.

3 Collected Data

In order to achieve an appropriate precision with supervised methods we need a big amount of training data. Creating this database manually is a hard and time-consuming task. In many cases it is hard even for humans to determine the polarity of a message, for instance:

> After a whole 5 hours away from work, I get to go back again, I'm so lucky!

In the above tweet we cannot decide precisely the polarity because the writer can be serious or just sarcastic.

In order to increase the size of the training data we acquired additional tweets, which we used in the unconstrained run for *Task–B*. We created an application which downloads tweets using the Twitter Streaming API. The API supports language filtering, which was used to get rid of non-English messages. Our manual investigations of the downloaded tweets revealed, however, that this filter allows a big amount of non-English tweets, which is probably due to the fact that some Twitter users did not set their language. We used Twitter4J[2] API (which is a Java library for the Twitter API) for downloading these tweets. We automatically annotated the downloaded tweets using the *Twitter Sentiment*[3] web application, similar to Barbosa and Feng (2010) but we used only one annotator. This web application also supports language detection, but after this extra filtration, our dataset still contained a considerable amount of non-English messages. After 16 hours of data collection we got $350,000$ annotated tweets, where the distribution of neutral, positive and negative classes was approximately 60%, 20%, 20%, respectively. For further testing purposes we have chosen $10,000$ tweets from each class.

4 Results

We report results on the two official test sets of the shared task. The "twitter" test set consists of $3,813$ tweets while the "sms" set consists of $2,094$ sms messages. We evaluated both test databases in two ways, in the so-called constrained run we only used the official training database, while in the unconstrained run we also used a part of the additional data, which was mentioned in the 3 section. The official training database contained $4,028$ positive, $1,655$ negative and $3,821$ neutral tweets while for the unconstrained run we used an additional $10,000$ tweets from each class. This way in each phase we got four kinds of runs, which were evaluated with the Naïve Bayes and Maximum Entropy classifiers.

In Table 1 the evaluation of the unigram-based model with the Naïve Bayes learner can be seen. The table contains the F-scores for the positive, negative and neutral labels for each of the four runs. The *avg* column contains the average F-score for the positive and negative labels, which was the official evaluation metric for *SemEval-2013 Task 2*. We got the best scores for the neutral label whilst the worst scores are obtained for the negative label, which is due to the fact that there were much less negative instances in the training database. It can be seen that the F-scores for the unconstrained run are better both for the tweet and sms test databases. For the unigram-based model the F-scores are higher when we used the Maximum Entropy model (see Table 2).

	pos	neg	neut	avg
twitter-constrained	0.59	0.09	0.65	0.34
twitter-unconstrained	0.60	0.17	0.65	0.38
sms-constrained	0.46	0.16	0.63	0.31
sms-unconstrained	0.47	0.38	0.53	0.42

Table 1: Unigram-based model, Naïve Bayes learner

	pos	neg	neut	avg
twitter-constrained	0.60	0.33	0.67	0.46
twitter-unconstrained	0.60	0.40	0.66	0.50
sms-constrained	0.47	0.31	0.69	0.39
sms-unconstrained	0.52	0.47	0.66	0.49

Table 2: Unigram-based model, Maximum Entropy learner

In Tables 3 and 4 the evaluation results can be seen for the normalized model. The normalization

[2] http://twitter4j.org

[3] http://www.sentiment140.com

step increased the precision for both learning algorithms and the Maximum Entropy learner is still better than Naïve Bayes. Besides this we noticed that for both learners in the case of the tweet test database, the unconstrained run had lower scores than the constrained whilst in the case of the sms test database this phenomenon did not appear.

	pos	neg	neut	avg
twitter-constrained	0.65	0.32	0.67	0.48
twitter-unconstrained	0.62	0.21	0.63	0.41
sms-constrained	0.56	0.27	0.72	0.41
sms-unconstrained	0.52	0.35	0.66	0.43

Table 3: Normalized model, Naïve Bayes learner

	pos	neg	neut	avg
twitter-constrained	0.66	0.40	0.68	0.53
twitter-unconstrained	0.61	0.42	0.64	0.51
sms-constrained	0.61	0.38	0.77	0.49
sms-unconstrained	0.57	0.47	0.72	0.52

Table 4: Normalized model, Maximum Entropy learner

The evaluation results of the feature-based model can be seen in Tables 5 and 6. In the case of the Naïve Bayes learner, the features did not increase the F-scores, only for the sms-unconstrained run. For the other runs the achieved scores decreased. In the case of the Maximum Entropy learner the features increased the F-scores, slightly for the constrained runs and a bit more for the unconstrained runs.

From this analysis we can conclude that the normalization of the messages yielded a considerable increase in the F-scores. We discussed above that this step also significantly reduced the size of the dictionary. The features increased the precision too, especially for the unconstrained run. This means that these features represent properties which are useful for those training data which are not from the same corpus as the test messages. We compared two machine learning algorithms and from the results we concluded that the Maximum Entropy learner performs better than the Naïve Bayes on this task. Our experiments also showed that the external, automatically labeled training database helped only in the classification of sms messages. This is due to the fact that the smses and our external database are from a different distribution than the official tweet database.

	pos	neg	neut	avg
twitter-constrained	0.65	0.32	0.67	0.48
twitter-unconstrained	0.62	0.17	0.79	0.39
sms-constrained	0.56	0.38	0.74	0.47
sms-unconstrained	0.54	0.29	0.70	0.41

Table 5: Feature-based model, Naïve Bayes learner

	pos	neg	neut	avg
twitter-constrained	0.66	0.41	0.69	0.54
twitter-unconstrained	0.63	0.43	0.65	0.53
sms-constrained	0.62	0.39	0.79	0.50
sms-unconstrained	0.61	0.49	0.75	0.55

Table 6: Feature-based model, Maximum Entropy learner

5 Conclusions and Future Work

Recently, sentiment analysis on Twitter messages has gained a lot of attention due to the huge amount of Twitter users and their tweets. In this paper we examined different methods for classifying Twitter and sms messages. We proposed special features which characterize the polarity of the messages and we concluded that due to the informality (slang, spelling mistakes, etc.) of the messages it is crucial to normalize them properly.

In the future, we plan to investigate the utility of relations between Twitter users and between their tweets and we are interested in topic-dependent sentiment analysis.

Acknowledgments

This work was supported in part by the European Union and the European Social Fund through project FuturICT.hu (grant no.: TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

References

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment Analysis

of Twitter Data. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 30–38, June.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on 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 *Poster volume*, Coling 2010, pages 36–44, August.

Albert Bifet and Eibe Frank. 2010. Sentiment Knowledge Discovery in Twitter Streaming Data.

Bernard J. Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. 2009. Twitter Power: Tweets as Electronic Word of Mouth. In *Journal of the American society for information science and technology*, pages 2169–2188.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter Sentiment Classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 151–160, June.

Bing Liu. 2010. Sentiment Analysis and Subjectivity. In N. Indurkhya and F. J. Damerau, editors, *Handbook of Natural Language Processing*.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

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*, May.

Erik Tjong Kim Sang and Johan Bos. 2012. Predicting the 2011 Dutch Senate Election Results with Twitter. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 53–60, April.

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.