SWATAC: A Sentiment Analyzer using One-Vs-Rest Logistic Regression

Yousef Alhessi and Richard Wicentowski
Swarthmore College
Swarthmore, PA 19081 USA
{yalhess1, richardw}@cs.swarthmore.edu

Abstract

This paper describes SWATAC, a system built for SemEval-2015's Task 10 Subtask B, namely the Message Polarity Classification Task. Given a tweet, the system classifies the sentiment as either positive, negative, or neutral. Several preprocessing tasks such as negation detection, spell checking, and tokenization are performed to enhance lexical information. The features are then augmented with external sentiment lexicons. Classification is done with Logistic Regression using a one-vs-rest configuration. For the test runs, the system was trained using only the provided training tweets. The classifier was successful, with an F1 score of 58.43 on the official 2015 test data, and an F1 score of 66.64 on the Twitter 2014 progress data.

1 Introduction

Since 2006, Twitter has grown into a ubiquitous global social platform. Millions of users compose Twitter messages, which are known as "tweets", to express their opinions and sentiments about the world around them. These tweets turn into valuable resources for sentiment analysis, a field that focuses on analyzing the attitude of speakers or writers towards a certain topic. Working with this informal text genre opens up a new realm of challenges in the natural language processing world. This paper describes a tweet sentiment classifier which has been applied to Subtask B of SemEval-2015 Task 10 (Rosenthal et al., 2015). The tweets generated by users contain Internet slang, unconventional punctuation and spelling, and typos, which require a different set of preprocessing tools than traditional genres like newswire text.

After preprocessing the tweets, classifying them into categories of positive, negative, and neutral presents another challenge. Many sentiment applications make use of lexicons to supply features to the system, populating a list of positive and negative types. Some publicly available sources include the MPQA Subjectivity Lexicon (Wilson et al., 2005), the Opinion Lexicon (Liu et al., 2005), and the Sentiment140 Lexicon (Mohammad et al., 2013). While some of these lexicons do not target tweets as their analysis subject, they each provide a mapping from n-grams to sentiment labels, which proves to be helpful in building our tweet sentiment analyzer.

After preprocessing, the system performs the classification task. The classifier we use is a one-vs-rest logistic regression classifier, so the system uses three binary classifiers: positive/not-positive, negative/not-negative, and neutral/not-neutral. The classifier also over-samples the low-frequency classes, learning from the same number of examples of each class overall.

The accompanying sections of the papers are organized as follows: Section 2 describes resources such as the lexicons used in the system. It also outlines the system design and the APIs that the system adopts. Section 3 describes the test runs and evaluates the system. Section 4 concludes the paper.

2 System Details

The main objective of our system is to determine if a tweet conveys a positive, negative, or neutral sentiment. To achieve this goal, the system first employs some preprocessing tools to enhance the lexical information. Then it relies on various sentiment lexicons to help with the classification of sentiments. For preprocessing, the system performs case-folding, detects negation, optionally uses a spell checker, performs tokenization, and makes use of unigrams, bigrams, and pairs of n-grams.

In addition to features extracted from the tweets, the system relies on four external sentiment lexicons. Three of them are pre-existing resources: the MPQA Subjectivity Lexicon (Wilson et al., 2005), the Opinion Lexicon (Liu et al., 2005), and the Sentiment140 Lexicon (Mohammad et al., 2013). The final lexicon is a manually created Emoji lexicon compiled by the authors.

After extracting features, a Logistics Regression classifier using a one-vs-rest setup is used to label each of the tweets.

2.1 Preprocessing

2.1.1 Case Folding

We use case folding to make every letter of every word in both the training and the test data lowercase. This helps in dimensionality reduction.

2.1.2 Negation Detector

The system includes a negation detector. Similar to (Pang et al., 2002), in this detector, we append a negation suffix to words that occur within a negation window between a negation key word and some punctuation. For example, the word "great", which is considered a positive word, is treated and learned as a different token if it is preceded by "not" as in "this pasta is not very great". This sentence would become "this pasta is not NOT_very NOT_great".

2.1.3 Jazzy

Jazzy is the Java Open Source Spell Checker[1]. Previous work had shown Jazzy to be effective (Miura et al., 2014). Though this was used during the development of the system, time constraints didn't allow its use in the final submission. Using five-fold crossvalidation, including Jazzy improved performance slightly, from an F1 score of 63.8 to 64.75.

[1]http://jazzy.sourceforge.net/

2.1.4 Twokenizer

Twokenizer is a tokenizer designed specifically for tweets (Gimpel et al., 2011). Twokenizer properly handles the tokenization of tweets without mangling URLs, mentions, or hashtags.

2.2 Sentiment Lexicons

2.2.1 MPQA

We make use of the MPQA Subjectivity Lexicon (Wilson et al., 2005). The lexicon is generated from the MPQA Opinion Corpus, which incorporates a wide range of news articles manually annotated for opinions and other private states. Although the MPQA lexicon list mainly targets news articles, it improved our system's classifications. The MPQA subjectivity lexicon provides a list of words with both their polarity (positive, negative, and neutral) and their strength (strong subjective, weak subjective). Our system made use of the polarity, but not the strength.

2.2.2 Opinion Lexicon

The Opinion Lexicon provided by Liu et al. (2005) consists of a list of positive words and a list of negative words. Because the lexicon is automatically generated from social media content, it contains misspelled lemmas, which could be beneficial to tweet analysis as tweets tend to include erroneous spellings and Internet slang (Liu, 2010). For example, we can find both words "awesome" and "awsome" in the list of positive words. In the negative list, we find "awful" as well as "aweful".

2.2.3 Sentiment140 Lexicon

The Sentiment140-Lexicon is a list of features with associations to positive and negative sentiments (Mohammad et al., 2013). The lexicon was created from the automatically-labeled sentiment140 corpus of 1.6 million tweets. The labeled features are unigrams, bigrams, and pairs of n-grams (unigrams-unigrams, unigrams-bigrams, bigrams-unigrams, and bigrams-bigrams). For example, some of the features we could see in the list are: the unigrams "@jeffery_donovan" and "xoxoxo", the bigrams "yeh yeh" and "praise !", and the pairs “done—had”, “i—, drinking”, “thank you—lovely”, and “good morning—can be”. Each feature has a score that reflects how positive or negative the feature is. If the word was seen in more positive contexts than negative contexts, it’s score is positive. The magnitude of the score is highest when the distribution is overwhelmingly positive, and the magnitude is closest to zero when the word appears equally in both positive and negative contexts. Negative words are scored similarly using negative values instead of positive values.

2.2.4 Emoji Lexicon

Our system uses a hand-created Emoji dictionary comprised of 16 positive[2] and 7 negative[3] emoticons. Only the most common Emoji in the training set were added to the lexicon. However, we chose to some exclude some emoticons because they portray a wide range of sentiments. For example, emoticons like ":-|" and ":|" were seen in both neutral and negative tweets. Using this specific set of emoticons improved the results when using cross-validation from an F1 score of around 62.5 to 64.8. A more extensive list might improve results, but given the time constraints, these 23 emoticons covered the test set adequately.

2.3 Classifier

Our system uses a one-vs-rest logistic regression classifier to analyze the sentiment of each tweet. Before the tweets get passed to the classifier, an oversampling process takes place to ensure equal numbers of each sentiment class during training. The classifier uses a one-vs-rest scheme, breaking down the classification process into three tasks: positive, negative, and neutral. Our classification task assumes that each sample is assigned to one and only one label.

2.3.1 One-Vs-Rest

We use a one-vs-rest strategy, building a classifier for each sentiment label (Hong and Cho, 2008). This means our system is comprised of three classifiers: positive/not-positive, negative/not-negative, and neutral/not-neutral. For each classifier, the class is compared against all the other classes. In other words, the features are screened to determine if they are positive, negative, or neutral in three separate stages: positive vs. non-positive, negative vs. non-negative, and neutral vs. non-neutral.

During testing, each instance is labeled by each of the three classifiers. When determining the label for a test instance, we would ideally like to have only one of the binary classifiers find a match. This usually happens when a tweet has many features expressing the same sentiment. However, when a tweet has contradicting features, the classifiers may contradict each either, either finding no matching class, or having multiple classifiers match a class. In cases of uncertainty, we use the labeling returned by the classifier with the highest confidence. Removing the one-vs-rest strategy decreases the score on cross-validation from 64.8 to 64.0.

2.3.2 Oversampling

In our classifier, we over-sample classes according to the number of examples we have in the training data. This means no matter what the distribution of our underlying training data is, the system learns from an equal number of examples of each class label. For example, if we have 100 negative instances in the training data and 200 non-negative instances, the negative instances would be sampled twice, whereas every non-negative example would be sampled only once. This way, a negative feature that is seen once is twice as strong or informative to our system as a non-negative feature that is seen once, and it would have the same weight as a non-negative feature that had been seen twice. This method decreased the system’s bias towards positive features. Removing oversampling decreases the score on cross-validation from 64.8 to 62.3.

2.3.3 Logistic Regression Model

The system uses the scikit-learn (Pedregosa et al., 2011) implementation of a Logistic Regression classifier. In this system, we use a simple logistic regression, where the model has one nominal variable (a class or non-class), and the features are used as measurement variables.

3 Test Runs

The final classifier included in the submitted system is an L2 regularized logistic regression algo-

[2]Positive :) :D :-) :-D :] :-] :’) :’-) ;) =) (: ;-) XD =D =] ;D

[3]Negative :(:-(:[:-[=(=/ :/

System	Live Journal 2014	SMS 2013	Twitter 2013	Twitter 2014	Twitter 2014 Sarcasm	Twitter 2015
SWATAC	68.67	61.30	65.86	66.64	39.45	58.43
Webis	71.64	63.92	68.49	70.86	49.33	64.84
Splusplus	75.34	67.16	72.80	74.42	42.86	63.73
Average	68.13	60.21	63.88	64.90	47.06	57.13

Table 1: Official results comparing the SWATAC system to the best performing systems on the Twitter 2015 and Twitter 2014 datasets, as well as the average performance on each dataset.

rithm, with a C value (the inverse of regularization strength) set to 1, and the tolerance for stopping criteria set to 0.0001, which are the default values provided by the scikit-learn library (Pedregosa et al., 2011). This system is stochastic and returns slightly different labellings on each run. Using five-fold cross-validation, the final classifier had an F1 score between 64.0 and 65.0.

The official results for our system are in Table 1. Our system has successfully scored a better than average F1 in all of the test sets, except for Twitter 2014 Sarcasm dataset. The table compares our system to two other submitted systems: Webis, the best scoring system on the Twitter 2015 dataset, Splusplus, the best scoring system on the Twitter 2014 progress test data, as well as the average scores of all submitted systems in each test data set.

4 Conclusion

This paper describes our submission to SemEval-2015's Task 10 subtask B. Our system uses several preprocessing tools, which includes case folding, negation, and tokenization. Several sentiment lexicons and a manually created Emoji lexicon are employed to help with classifying message polarities. The system uses a logistic regression classifier along with a one-vs-rest scheme to perform a three-stage classification. The results indicate that our system generally performs well, with an F1 score of 58.43 on the 2015 test data.

References

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 HLT '11: Short Papers*, volume 2, pages 42–47.

Jin-Hyuk Hong and Sung-Bae Cho. 2008. A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. *Neurocomputing*, 71(16-18):3275–3281.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the Web. In *WWW '05*, pages 342–351.

Bing Liu. 2010. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:627–666.

Yasuhide Miura, Shigeyuki Sakaki, Keigo Hattori, and Tomoko Ohkuma. 2014. TeamX: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In *Proceedings of SemEval-2014*, pages 628–632.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of SemEval-2013*, pages 321–327.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP '02*, pages 79–86.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. 2015. Semeval-2015 task 10: Sentiment analysis in Twitter. In *Proceedings of SemEval-2015*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HTL-EMNLP '05*, pages 347–354.