INSRT –INterpreted Sentiment in Real Time

Michael Haas,Tilman Wittl

May 8, 2013

1 The Project: INSRT

The INSRT project - short for INterpreted Sentiment in Real Time - performs sentiment polarity classification on German tweets in real time. Classification of sentiment polarity in social media has become an important tool for reputation monitoring and trend analysis. For INSRT, we monitor the list of currently trending topics provided by the Twitter microblogging service and classify the tweets concerning these topics into positive, negative or neutral sentiment polarity. An user-friendly web interface is provided to show monitoring results; that is, a list of trending topics, associated tweets and sentiment classification results.

2 The INSRT approach

We use a machine learning approach to train a sentiment classifier. We have been collecting training data from the Twitter streaming API for several weeks. Tweets are labeled based on the presence of emoticon tokens: *positive* for tweets with a positive emoticon such as **:)**, *negative* for tweets with a negative emoticon such as **:(**, and *neutral* for tweets containing both categories. Our approach is roughly inspired by Narr et al. (2012), who describe an approach to language-independent tweet sentiment classification. Their heuristic for labeling is based on emoticon lists. From these noisily labeled tweets, a naive bayes classifier is trained on token n-gram features. In comparison, we choose slightly different features and add a sentiment lexicon for the German language to try and improve classifier accuracy. We will now present our processing pipeline, an evaluation of our classifier models and our live monitoring.

3 Preprocessing

In order to create training instances from the previously crawled Tweets to train our machine learning approach and to classify the Tweets later, several preprocessing steps are necessary. To split tweets into individual words, we use the tokenizer included in the TweetNLP–Tools. Gimpel et al. (2011) This tokenizer is based on regular expressions to split the tweets into words and is aware of domain-specific language language like emoticons.
The next step in our preprocessing pipeline is a kind of part of speech tagging. For this task task we use the TweetNLP–Tools again. As we are not interested in giving each word a fine grained POS tag, we simply modify the TweetNLP–POS-Tagger to label each word as *word* (Tag: W) or *emoticon* (Tag: E).
After tokenization and part of speech tagging we use two ressources to assign a sentiment label (positive, neutral or positive) to both emoticons and words. For the emoticons we use a handcrafted list of the common emoticons. For the words we use the GermanPolarityClues sentiment lexicon presented by Waltinger (2010). This lexicon provides a list of about 10.000 lemmas annotated with sentiment labels. We load the lexicon into a prefix trie data structure to allow fast lookups. As the lexicon provides lemmas this data structure has another advantage, due to its internal structure an inflected word can also be looked up to some extent.

4 Features

The following features are extracted from preprocessed tweets.

Language as detected by the LDIG [1] language detector

Token count based on TweetNLP–Tools tokenization

Normalized Lexicon Sentiment Score sum of positive and negative word occurences in a tweet divided by tweet length.

Emoticon counts for positive, negative and neutral smileys

Lexicon Sentiment label *positive* for tweets with positive normalized sentiment scores, *negative* for negative normalized sentiment scores, *neutral* otherwise

Repeated characters *Yes* if a character is repeated three times, e.g. "looool"

[1](c)2011-2012 Nakatani Shuyo / Cybozu Labs Inc., https://github.com/shuyo/ldig

Repeated uppercase character *Yes* if three characters in a row are uppercase, e.g. "OMG"

Word Vector occurence counts in tweet for 1000

N.B: The *emoticon counts* features are used to assign noisy labels to tweets as explained earlier.

5 Evaluation

We run two experiments. The first experiment aims to compare our implementation against the approach presented by Narr et al. (2012). In their setup, they disregard neutral tweets and balance the classes so that an equal number of positive and negative tweets are used for training. Notably, the classes are not balanced in the test set. We start with the **basic** feature set including only *word vector*, *hashtag vector*, *repeated characters*, *repeated uppercase characters*. Emoticon tokens used to generate the noisy labels are removed from the feature set to prevent the classifier from developing a bias for these features.
In a second run, we add the *lexicon sentiment label* feature to the **basic** feature set to form the **lexicon** data set.

The second experiment aims to evaluate our full application setting where tweets have to be classified into positive, negative and neutral.
The **basic** and **lexicon** feature sets are used as described above. We add a third run based on **basic** named **emoticon** which adds the emoticon tokens back into *word vector* feature. A fourth run which includes the **full** feature set described in the feature list as well as the emoticon tokens excluded earlier.

Tweets were acquired from the Twitter Streaming API. The "filter" parameter for the API was to used to select tweets originating from within Germany. The "sample" API endpoint was used. Tweets were collected for several weeks in Q1 and Q2 2013.

Further filtering is necessary as many tweets are in languages other than German. All tweets not labeled as German by the LDIG language classifier are removed. Approximately 140.000 tweets remain.

During development, we used the annotated data set created by Cui et al. (2011)[2] to tune the classifiers. Our test set is the *agree-2* data set provided by Narr et al. (2012) which consists of 1660 annotated tweets. We use the

[2]Many thanks to Anqi Cui for sending us this data set via email.

Table 1: Evaluation: Positive, negative classification

Classifier	Accuracy
Baseline	0.60
Naive Bayes – basic	0.62
BayesNet – basic	0.42
Naive Bayes – lexicon	0.73
BayesNet – lexicon	0.73

Table 2: Evaluation: Full classification

Classifier	Accuracy
Baseline	0.64
Naive Bayes – basic	0.55
BayesNet – basic	0.65
Naive Bayes – lexicon	0.61
BayesNet – lexicon	0.66
Naive Bayes – emoticon	0.60
BayesNet – emoticon	0.67
Naive Bayes – full	0.65
BayesNet – full	0.69

Weka machine learning package (Hall et al. (2009), version 3.7.7) with default settings.

For all experiments, the baseline is to assign the majority class found in the test set. Figures are rounded to two digits.

5.1 Evaluation: Narr et al

For the training set, approximately 29000 tweets remain after disregarding all neutral tweets. 14% of these tweets are negative. We resample the data without replacement to generate an uniform class distribution, after which 4081 training instances remain. All results are shown in table 1.

5.2 Evaluation: Full application setting

The full training set consisting of approximately 140000 German tweets is used. We provide results for the basic feature set, basic + *lexicon sentiment label*, basic + emoticon tokens in *word vector* and for the full feature set.

All results are shown in Table 2.

6 Live Monitoring

Twitter is a dynamic sort of media. Therefore we provide an application to monitor current trending topics and their Tweets live. We provide a small website[3] which displays this information.
Internally our application is built with the Mojolicious::Lite framework [4]. We use the Twitter API to check every 2 minutes for new trending topics. Using this list of topics we connect to the Twitter Streaming API to grab the actual Tweets matching these topics. These Tweets are then processed as described in section *Preprocessing* and classified as presented above. Finally the tweets and the results are shown on a website.

7 Discussion & Results

We set out to build a live sentiment monitoring tool for twitter trending topics specifically for the German language. Our project is live and usable at `http://insrt.vela.uberspace.de/app/`.
We evaluate our model on a previously existing data set, outperforming the majority class baseline with an accuracy of 73%.
Adding either emoticon tokens or lexicon-based sentiment information can improve classifier performance, as seen in 2 and 1. Best performance is achieved with the full feature set. As increased amounts of training data improve performance according to Narr et al. (2012), the impact of adding lexicon-based information might be reduced as the model better learns associations between words and sentiments.
The original approach presented by Narr et al. (2012) reaches 79.8% without relying on language-specific ressources. This discrepancy in performance is likely explained by our feature selection: our approach currently only uses the 1000 most frequent tokens counted across all tweets and thus across all languages in the corpus. Given that natural language follows a Zipf distribution, we likely miss many relevant tokens conveying sentiment. These are accounted for by Narr et al. (2012) as they only eleminate tokens with a count of less than 2.
Other reasons might include different emoticon lists and training set sizes, which was smaller in our case: 4000 tweets (due to resampling) vs 49000 tweets.
As a side remark: the development set provided by Cui et al. (2011) shows worse accuracy values than the test set provided by Narr et al. (2012), with

[3]http://insrt.vela.uberspace.de/app/
[4]http://mojolicio.us/

values around 49% being typical when classified using the model for **full** task. On the other hand, we achieve 49% accuracy when predicting the actual label with the gold label, i.e. the noisy label and the gold label agree in 49% of all cases. We make a similar observation for the training set, where agreement between noisy labels and gold labels occurs in 68% of all cases. This is close to the performance obtained by the BayesNet classifier (69%) as seen in table 2. This agreement between noisy labels and gold labels can be seen as another baseline for classifier performance. This raises the question if quality of the noisy labels is sufficiently high to facilitate learning. Another question is whether the classifier generalizes well. This issue seems less important for the original *positive, negative* classification task: by defaulting *neutral* noisy labels to the *positive* majority class and comparing noisy labels to gold labels, we achieve 63% agreement. Thus, we see a clear improvement in our models at 73% accuracy (table 1) for this particular baseline.

References

Anqi Cui, Min Zhang, Yiqun Liu, und Shaoping Ma. Emotion tokens: Bridging the gap among multilingual twitter sentiment analysis. In Mohamed-VallMohamed Salem, Khaled Shaalan, Farhad Oroumchian, Azadeh Shakery, und Halim Khelalfa, editors, *Information Retrieval Technology*, volume 7097 of *Lecture Notes in Computer Science*, pages 238–249. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-25630-1.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, und Noah A. Smith. 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, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-88-6.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, und Ian H. Witten. The weka data mining software: An update. volume 11. 2009.

Sascha Narr, Michael Hülfenhaus, und Sahin Albayrak. Language-independent twitter sentiment analysis. 2012.

Ulli Waltinger. Germanpolarityclues: A lexical resource for german sentiment analysis. In *Proceedings of the Seventh International Conference on*

Language Resources and Evaluation (LREC), Valletta, Malta, May 2010. electronic proceedings.