KLUEless: Polarity Classification and Association

Nataliia Plotnikova and **Micha Kohl** and **Kevin Volkert** and **Andreas Lerner**
and **Natalie Dykes** and **Heiko Ermer** and **Stefan Evert**
Professur für Korpuslinguistik
Friedrich-Alexander-Universität Erlangen-Nürnberg
Bismarckstr. 6, 91054 Erlangen, Germany
{nataliia.plotnikova, micha.kohl, kevin.volkert, andreas.lerner,
natalie.dykes, heiko.ermer, stefan.evert}@fau.de

Abstract

This paper describes the KLUEless system which participated in the SemEval-2015 task on "Sentiment Analysis in Twitter". This year the updated system based on the developments for the same task in 2014 (Evert et al., 2014) and 2013 (Proisl et al., 2013) participated in all five subtasks. The paper gives an overview of the core features extended by different additional features and parameters required for individual subtasks. Experiments carried out after the evaluation period on the test dataset 2015 with the gold standard available are integrated into each subtask to explain the submitted feature selection.

1 Introduction

The SemEval-2015 shared task on "Sentiment Analysis in Twitter" (Rosenthal et al., 2015) is a rerun of the shared task from SemEval-2014 (Rosenthal et al., 2014) with three new subtasks. While subtasks A and B were identical to the tasks of SemEval-2014 and dealt with the identification of polarity in a given message, subtask C, D and E were new. In subtask C a topic was given, towards which the sentiment in a message had to be identified. Subtask D was similar to subtask C, as the sentiment towards a given topic had to be identified, but in this subtask several messages were given from which the sentiment had to be drawn. Ultimately in subtask E, the sentiment of a given word or phrase had to be measured on a score ranging [0, 1], indicating its association with positive sentiment.

The training data for subtasks A and B are the same as in SemEval-2014 (Rosenthal et al., 2014) and SemEval-2013 (Nakov et al., 2013). For subtask A, there are 9,505 training items with 6,769 items in development set and 3,912 items in the test set. For subtask B, there are 10,239 training items, 5,907 items in the development set and 3,861 in the test set. For subtasks C and D the same training sets as for subtasks A and B were used by our team. A pilot task E aimed at evaluation of automatic methods of generating sentiment lexicons had no training set, a detailed approach used for this subtask will be given in Section 3.

This paper describes the updated system with our efforts to improve it after the evaluation period. The KLUEless system was ranked within the top 3 participants to subtasks A (rank 2 out of 11), C (rank 2 out of 7) and D (best result out of 6 teams). It scored 5th place in subtask E, but only 13th place in subtask B (rank 13 out of 40 teams). In the following chapters, we will describe the way KLUEless dealt with the tasks stated and our results for these tasks.

2 The KLUEless Approach

The KLUEless polarity classifier is an updated version of the SentiKLUE system used for the SemEval-2014 shared task on "Sentiment Analysis in Twitter" (Evert et al., 2014) which in its turn was based on the KLUE system that participated in the SemEval-2013 task for sentiment analysis of tweets (Proisl et al., 2013). Maximum Entropy (known as Logistic Regression in the implementations of the Python library scikit-learn[1] (Pedregosa et al., 2011))

[1] http://scikit-learn.org

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 619–625,
Denver, Colorado, June 4-5, 2015.

is a machine learning algorithm in the submission for all subtasks (A-D). The detailed overview of all features used by the system is given in the previous papers. This section is a brief summary of the old features extended by the new set of features that the system extracted from the training data for subtasks A,B,C, and D. The old feature vectors taken by the system as input are:

1) the sum of positive and negative scores over all words of each message as well as an average polarity score per tweet. The scores are taken from 8 different sentiment lexicons (AFINN [2], MPQA[3], SentiWords[4], Sentiment140 (both bigrams and unigrams) [5], NRC Hashtag Sentiment Lexicon (both bigrams and unigrams) with numeric polarity scores extended with lists of distributionally similar words based on the AFINN sentiment lexicon (Proisl et al., 2013, Sec. 2.2).

2) counts of positive and negative emoticons based on the list of 212 emoticons and 95 internet slag abbreviations from Wikipedia classified manually as negative (-1), neutral (0) or positive (1) (Proisl et al., 2013, Sec. 2.3).

3) a bag-of-words model with word ngrams (unigrams and bigrams) occurring in at least 2 different messages for subtask A and in 3 different messages for subtask B, C and D.

4) a negation heuristic inverting the polarity score of the first sentiment word within 4 tokens after a negation marker. In the bag-of-words representation the following 4 tokens after a negation are prefixed with not_.

The new feature set added to the old one encompasses the following new features:

5) a number of question marks in a message,

6) a number of exclamation marks,

7) a number of combinations of "!?",

8) a number of letters in upper case,

9) presence or absence of elongated vowels occurring more than twice,

10) automatically generated lexicons described in Section 3 which were left out in the submission, though used in the development phase.

These features form the core system. The features specific to subtasks A and B are described in their corresponding subsections below.

3 Creating Sentiment Lexica

3.1 Subtask E

For Subtask E, we collected Twitter data for automatic annotation and subsequent score computation for individual target terms. A similar approach was suggested last year (Kiritchenko et al., 2014). Our tweet collection was built mostly by filtering the English Twitter Streaming API for target terms provided in the test data using a Python script based on code from Russell (2014). The downloaded tweet texts were stripped of retweet boilerplate and usernames and URLs were replaced with anonymous placeholders. Redundant tweets and tweets containing no useful information (e.g. no English words) were discarded, resulting in a total of about 6.5 million.

We used three sources to annotate our tweet data. One was our main KLUEless system, assigning either positive, negative or neutral sentiment to a tweet. The other two were manually annotated lists of 328 hashtags (manually selected and reannotated from a lexicon generated by Mohammad et al. (2013)) and 67 emoticons (manually selected from a list generated from wikipedia articles[6,7]). Tweets were tagged positive when they contained at least one positive and no negative hashtag or emoticon respectively and vice versa.

Because annotation based on hashtags and emoticons showed promising results on the test data and because we wanted to rely as little as possible on existing sentiment lexica that greatly influence the annotations provided by our KLUEless system, we gave priority to hashtag and emoticon based sentiments in this order and fell back to KLUEless annotations if either no other information was available or the available information was conflicting. This overall sentiment annotation also allowed for tweets to be tagged as neutral as this was a possible output from the KLUEless annotation.

To counter data sparsity, a back-off approach relying on large scale word clusters based on twitter

[2] http://www2.imm.dtu.dk/pubdb/p.php?6010

[3] http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

[4] https://hlt.fbk.eu/technologies/sentiwords

[5] http://www.umiacs.umd.edu/ saif/WebPages/Abstracts/NRC-SentimentAnalysis.htm

[6] http://de.wikipedia.org/wiki/Emoticon

[7] http://en.wikipedia.org/wiki/Emoticons_(Unicode_block)

data (Owoputi et al., 2012) was introduced. The frequency information of any target term occurring below a set frequency threshold t_f was replaced by combined frequency information from cluster members. In order to exclude marginal cluster members, only those members that together made up a set proportion t_c of the original cluster data were used. So, if back-off was applied for the term *okayyy* for example, and t_c was set to 0.8, the combined frequency information of the terms *ok*, *okay* and *alright*, which are the three most frequent cluster members that make up 80% of all tokens in this cluster, would be used. We disabled back-off for hashtags as the cluster data contained a considerably big cluster with arbitrary hashtags that would disrupt any positive effect of cluster based back-off for these cases. The final scores for the target terms

$$score = \frac{f_{pos}}{f_{pos} + f_{neg}} \quad (1)$$

Figure 1: Maximum likelihood scoring equation.

were computed using a simplistic maximum likelihood estimate based on their occurrences in positive and negative contexts (see Figure 1), ignoring information from tweets tagged as neutral. Multiple occurrences of the same term within one tweet were counted as one. Any terms that after cluster back-off still had no frequency information available were assigned a default score of 0.5. More sophisticated scoring systems based on extensions to this approach will be discussed in Section 8.

3.2 Lexica for Use with the KLUEless System

We applied a similar method for creating our own sentiment lexica for use with our main system. We used the same procedure described above for counting frequencies of uni- and bigrams in all data that was collected for subtask E trial and test runs (approximately 13 million tweets). Since there were no target terms for which cluster based back-off could be applied we implemented a workaround in order to still be able to remedy data sparseness.

By creating separate lexica for every application of our KLUEless system, we were able to use the trial and test data of any specific run as a target for back-off, effectively using all words found in the data of a given run as a list of target terms. This also enabled us to filter out any terms that weren't useful for the specific run and create lexica that only contained relevant information. For missing unigrams, we tried to find the most frequent term in its cluster that also occurred in our tweet data and adopted its frequency data. For missing bigrams, we applied a more complex approach as the cluster data didn't contain information about bigrams. We set an arbitrary threshold of 10, assuming that any bigram occurring at least this frequently in the target data would probably not be a spelling error. For bigrams that occurred less often in the target data and not at all in the data used for collecting our frequency information we applied cluster-back off on a unigram level and tried to find a combination that also occurred in our tweet data.

After this process of filtering and back-off, we used the same simplistic scoring approach as before to generate separate uni- and bigram lexica for each submission run of our KLUEless system.

4 Task A: Contextual Polarity Disambiguation

Using the core system described in Section 2, we computed the features for the whole message and received three features with probabilities of being positive, negative and neutral for each complete tweet. In order to adjust the classifier to message parts, we added an additional feature to the core system, character ngrams. 1 to 5 characters were taken within word boundaries of a marked part of a message if it occurred at least 20 times. Using the extended classifier we computed the new set of features for marked parts of each message and added previously assigned class probabilities to feature vectors generated from corresponding full messages. The KLUEless system received its core feature vectors extended by ngrams and three class probabilities as input and generated final polarity labels to all marked parts of each message.

The specific features used improved the performance (see Table 1). Results for the submitted version is typeset in italics, the best result is typeset in bold.

The character ngrams improved the overall classifier performance for subtask A. The system achieved rank 2 out of 11 systems (with F-score 84.51). Inter-

features	F_{pos}	F_{neg}	F_{neut}	F_w	$F_{pos+neg}$	Acc
SentiKLUE	.8740	.7874	.0303	.7939	.8307	.8186
KLUEless						
+ ngrams$_{1..5}$	.8814	.8080	.1513	.8126	*.8451*	.8289
+ lexicon$_{2014B}$	.8829	.8155	.1513	.8160	**.8492**	.8321

Table 1: Evaluation results for subtask A on the test set 2015.

estingly, using automatically generated lexicon with tools developed for Task E for the training data of SemEval 2014 (Task B) could have improved the results bringing our system to the first place with F-score of 84.92 (best system: 84.79). As it was not evident on the development data, we have not included this lexicon when submitting the results. Trying to use this lexicon for other subtasks after the evaluation stage did not improve the scores. Therefore, it might be a coincidence.

5 Task B: Message Polarity Classification

The system scored 13th place out of 40 on subtask B with F-score 61.20 (best system: 64.84). As in subtask A, we used the basic feature set described in Section 2 extended by task specific features. We extended the initial bag-of-words model with trigrams occurring in at least 3 different messages. The large character ngrams generated from characters inside word boundaries only (padded with space on each side) were added to the feature vectors. Using the extended set of features KLUEless generated final polarity labels for test messages.

Results for the submitted version is typeset in italics, the best result is typeset in bold (see Table 2).

features	F_{pos}	F_{neg}	F_{neut}	F_w	$F_{pos+neg}$	Acc
SentiKLUE	.6618	.5348	.6731	.6471	.5983	.6448
KLUEless						
+ ngram$_{8..9}$	.6644	.5533	.6777	.6529	.6089	.6506
+ ngram$_{8..9}$ +						
+ trigrams	.6674	.5566	.6792	.6554	***.6120***	.6531

Table 2: Evaluation results for subtask B on the test set 2015.

8 and 9 characters inside word boundaries improved the overall total score both on the development set and on the test set 2015. The same positive influence was noticed for trigrams added to the bag-of-words model.

6 Task C: Topic-Based Polarity Classification

For the subtask C we used exactly the same approach used for subtask B. Therefore, we have ignored topics towards which sentiments were to be identified and assigned polarity labels generated by KLUEless to the full messages. Nevertheless, the system ranked 2 out of 7 teams with F-score 45.48 (best system: 50.51). The assigned labels were projected onto the list of test topics. The feature set for this subtask was extended as described in Section 5 since it is the best found configuration. For messages where both a positive and negative sentiment towards the topic are expressed, the stronger sentiment is chosen by the classifier.

7 Task D: Detecting Trends on a Topic

The task was in determining a dominant sentiment towards a target topic. Feature vectors based on the values listed in Section 2 were extracted from the 2,383 test sentences and processed by KLUEless. The classifier assigned numeric values in the range from 0 to 1, which corresponds to the probability of being positive, negative and neutral to each tweet. For each tweet the highest score was selected and its value was added to the total score of positive, negative or neutral values assigned to the tweets of the same topic. These triples were used to calculate the correlation between positive scores and the sum of positive and negative ones for each topic.

In the submitted version we made use of neutral values as well and ended up with the following formula for the sentiment score of a topic:

$$score = \frac{topic_{pos} + topic_{neut} * A/2}{topic_{pos} + topic_{neut} * A + topic_{neg}} \quad (2)$$

Figure 2: Sentiment score calculation.

where $topic_{pos}$ is a sum of all positive values of tweets on the same topic for which the highest value was positive. The same idea was used for $topic_{neut}$ and $topic_{neg}$. The factor A is a numeric value added to incorporate neutral tweets into the ratio of positive values to [positive + negative] values of tweets. This is the system we submitted with factor A set to 0.2 defined on experiments for the training data. The system performed best of all and achieved the

1st place out of 6 on the task.

After the evaluation stage, we tried to improve the performance and test the same approach with different parameters for factor A as well as without a factor at all using the test data with their gold standard set. The result for the submitted system is typeset in italics, the best result is in bold font in Table 3.

A = 0.0	A = 0.01	A = 0.1	A = 0.2	A = 0.8
0.1926	**0.1924**	0.1954	*0.2017*	0.2320

Table 3: Average absolute difference depending on factor A on the test set 2015.

8 Task E: Association of Terms with Positive Sentiment

For our submission, we set t_c to 0.8 and t_f to 0, effectively applying back-off only for terms that didn't occur in our data at all. We did not disable back-off for hash-tag terms as has been noted in section 3, a change which should have had little impact on the resulting score, as our submission relied on cluster information for only seven items in the target terms, only one of which was a hashtag. Our results were ranked 5th out of 10 participants for task 10 subtask E with a Spearman rank correlation coefficient of 0.766, which was to be expected on the basis of very similar results on the trial data with the same setup.

In the following, the effect of the applied back-off method based on clustering, the individual effects of its two parameters t_c and t_f as well as some experimental extensions for improving our score shall be discussed. Back-off for hashtag terms was disabled for all subsequent experiments.

	Spearman Correlation		
t_f	$t_c = 0.8$	$t_c = 0.6$	$t_c = 0.4$
-	0.767	0.767	0.767
0	0.766	0.767	0.767
20	0.765	0.765	0.766
100	0.751	0.751	0.752
200	0.742	0.742	0.742
500	0.722	0.722	0.720

Table 4: Results for different settings for frequency and cluster threshold parameters (t_f: frequency threshold for back-off, t_c: cluster proportion threshold).

8.1 Cluster Parameters

The first set of experiments was conducted to evaluate the effect of the two clustering parameters, the cluster proportion threshold t_c, which determines the proportion of cluster members that is used for collecting cluster information during frequency counting, and the frequency threshold t_f, which determines the maximum frequency of terms in our data to be affected by back-off.

The results in Table 4 show that, first of all, t_c seems to have only minimal effect on the final correlation score. This suggests that either a very small number of cluster members make up most of each cluster, minimizing the effect of different cut-off points, or that the clusters are in fact very homogeneous in their structure, at least for the majority of each cluster's members, resulting in similar frequency proportions for most of their members.

The second finding was that as more terms are affected by back-off with higher values for t_f, the score seems to get progressively worse. This is a somewhat unexpected result, as we were able to achieve some gains by using a frequency threshold of 100 on the trial data (after the deadline for subtask E), but is most likely due to the fact that our two tweet corpora are approximately the same size for both trial and test data, albeit the considerable difference in the number of target terms. The obvious consequence is data sparsity, resulting in much more terms being affected by back-off using the same threshold in the test run as compared to the trial run.

8.2 Extensions

A second set of experiments was based on three extensions to our basic approach. The first consists of add-λ smoothing, which adds a given number λ to all frequency counts, eliminating zero frequencies and generally smoothing frequency counts. Another extension was the inclusion of a method for bias correction. This means we assumed that the population contains a certain proportion b of positive tweets and adjusted the frequency counts obtained through our balanced sample to those expected under this bias assumption (the default assumption, where no correction is applied being of course 50%). The last extension was to adjust our frequency proportions

by computing binomial confidence intervals for a set confidence level c and replacing the actual proportions by conservative estimates (the lower end of the confidence interval for proportions over 50% and the upper end for those below). This results in an overall correction towards a balanced proportion and consequently in scores closer to the neutral 50% mark.

As general experiments with these extensions confirmed our findings of higher frequency thresholds for clustering worsening results, and cluster thresholds being of small importance, the systematic experiments discussed in the following were conducted with t_f set to zero, effectively applying back-off only for terms that didn't occur at all in our data and t_c set to 0.8, which is a configuration consistent with the settings used for submission. Experimenting with the proposed extensions led to rather discouraging results and a maximum improvement of 1.0% for the Spearman correlation score.

	Spearman Correlation	
b	$\lambda = 0$	$\lambda = 1$
0.6	0.763	0.768
0.5	0.766	0.768
0.4	0.768	0.768
0.3	0.767	0.768
0.2	0.762	0.768

Table 5: Results for different bias correction settings (b: assumed proportion of positive tweets in population).

Applying bias correction only led to a marginal improvement of 0,2% when b was set to 40%, add-one smoothing seemed to offset the negative effect of different proportion assumptions (see Table 5). Keeping bias correction at this setting and including conservative estimates based on confidence intervals had consistently negative effects, which were increased by add-one smoothing and minimized by a very low confidence level c of 0.1 (see Table 6). Surprisingly, another experiment including conservative estimates for this confidence level and different bias correction settings achieved an optimal result of 77.6% correlation with add-one smoothing and an assumed population proportion b of 0.1 positive tweets (see Table 7), which is of course a highly unlikely assumption.

		Spearman Correlation	
b	c	$\lambda = 0$	$\lambda = 1$
0.4	-	0.768	0.768
0.4	0.1	0.768	0.758
0.4	0.2	0.766	0.756
0.4	0.3	0.763	0.753

Table 6: Results for conservative estimates using different confidence levels (b: assumed proportion of positive tweets in population, c: confidence level for conservative estimates).

		Spearman Correlation	
b	c	$\lambda = 0$	$\lambda = 1$
0.4	-	0.768	0.768
0.6	0.1	0.752	0.743
0.3	0.1	0.775	0.767
0.2	0.1	0.773	0.773
0.1	0.1	0.760	0.776

Table 7: Results for conservative estimates using different bias correction settings (b: assumed proportion of positive tweets in population, c: confidence level for conservative estimates).

The results of all performed experiments seem to indicate that, while add-one smoothing and the proposed method of bias correction may provide opportunity for optimization, adjusting proportions with regard to conservative estimates using binomial confidence intervals seems to only show positive effects in combination with the other extensions. Intuition and the fact that these effects proved to be rather arbitrary suggest that no predictable effects seem possible and this third extension could only lead to a score improvement because of strong overtraining. The proposed back-off approach using cluster information has been shown to have exclusively negative effects, even when applied only to terms that didn't occur in our data at all. This can of course be said to be a matter of luck, depending on how close the gold standard labels for such terms are to a given default score that is assigned instead of the result of cluster based back-off. Further experiments should be conducted to evaluate whether this approach can be beneficial when applied to scores that are based on a larger data set.

9 Conclusion

The methods discussed in this paper are suited to the polarity classification in Twitter, our system ranking among the top systems for 3 out of 5 subtasks. In future, we would like to experiment with new fea-

tures for message polarity classification that can improve the prediction quality. We would also like to experiment with automatically generated lexica for new domains. Overall it can be assumed that our approach to determining association of terms with positive sentiment was most likely limited by data sparsity due to insufficient tweet data for our frequency counts. Collecting more tweet data, we will experiment with different methods involving add-λ smoothing and bias correction.

References

Stefan Evert, Thomas Proisl, Paul Greiner, and Besim Kabashi. 2014. SentiKLUE: Updating a polarity classifier in 48 hours. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 551–555, Dublin, Ireland, August.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M. Mohammad. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research (JAIR)*, 50:723–762.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013)*, Atlanta, Georgia, USA, June.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in Twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, Georgia, USA, June.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, and Nathan Schneider. 2012. Part-of-speech tagging for twitter: Word clusters and other advances. *School of Computer Science*.

Fabian Pedregosa, Gaël 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. In *Journal of Machine Learning Research*, volume 12, pages 2825–2830.

Thomas Proisl, Paul Greiner, Stefan Evert, and Besim Kabashi. 2013. KLUE: Simple and robust methods for polarity classification. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 395–401, Atlanta, Georgia, USA, June.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. Semeval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

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 the 9th International Workshop on Semantic Evaluation*, SemEval '2015, Denver, Colorado, June.

Matthew A. Russell, 2014. *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*, chapter 9.8. Sampling the Twitter Firehose with the Streaming API. O'Reilly, 2 edition.