VCU-TSA at Semeval-2016 Task 4: Sentiment Analysis in Twitter

Gerard Briones and **Kasun Amarasinghe** and **Bridget T. McInnes, PhD.**
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, 23284, USA
{brioinesgc, amarasinghek, btmcinnes}vcu.edu

Abstract

The aim of this paper is to produce a methodology for analyzing sentiments of selected Twitter messages, better known as Tweets. This project elaborates on two experiments carried out to analyze the sentiment of Tweets from SemEval-2016 Task 4 Subtask A and Subtask B. Our method is built from a simple unigram model baseline with three main feature enhancements incorporated into the model: 1) emoticon retention, 2) word stemming, and 3) token saliency calculation. Our results indicate an increase in classification accuracy with the addition of emoticon retention and word stemming, while token saliency shows mixed performance. These results elucidate a possible classification feature model that could aid in the sentiment analysis of Twitter feeds and other microblogging environments.

1 Introduction

Twitter is a widely used microblogging environment which serves as a medium to share opinions on various events and products. Because of this, analyzing Twitter has the potential to reveal opinions of the general public regarding these topics. However, mining the content of Twitter messages is a challenging task due to a multitude of reasons, such as the shortness of the posted content and the informal and unstructured nature of the language used. The aim of this study is to produce a methodology for analyzing sentiments of selected Twitter messages, better known as Tweets. This project elaborates on two experiments carried out to analyze the sentiment of Tweets, namely, Subtask A and Subtask B from SemEval-2016 Task 4 (Nakov et al., 2016).

Subtask A: Message Polarity Classification. The goal of this subtask was to predict a given Tweet's sentiment from three classes: 1) positive, 2) neutral, or 3) negative.

Subtask B: Tweet classification according to a two-point scale. The goal of this subtask was to classify a given Tweet's sentiment towards a given topic. The sentiments were limited to positive and negative, unlike Subtask A.

2 Method

We viewed both tasks as classification problems. We represented the Tweets in a statistical feature matrix and performed the classification using supervised machine learning classification algorithms. Several different feature vectors were experimented with and the same set of feature vectors were applied to both Subtask A and Subtask B.

2.1 Feature Vectors

Our methods consume Tweet data and output a matrix where each row represents a Tweet and each column represents a feature. The values in this matrix are the frequency of appearance of a feature in the Tweet. As reference for the rest of the project, please note that $n\text{-}grams$ are a continuous set of n terms in a document. Thus, when n=1, we are representing a unigram, or a single word; when n=2, we are representing a bigram, or a pair of words, and so on. We evaluated unigram, bigram and trigram models, but discuss only the unigram model. The bigram and trigram models results showed to be less effective.

2.1.1 Unigram Model

As a baseline, a unigram model was used as the primary feature vector. The unigram model consists of several one-state finite automatas, splitting the probabilities of different features in a context. The probability of occurrence for each feature is independent. In our project, each word in a Tweet represents a feature.

For the first step of creating the unigram feature vector, numbers and special characters were removed from each Tweet, since they carry little information when taken out of context. The Tweets were then converted to all lower case to reduce the dimensionality of the data, whereby different users' capitalization does not factor in as a new, separate feature. Next, the Tweets were tokenized by breaking up the messages into single word units that each represent a unique feature. All stop words were then removed from these token sets. Stop words, such as "the" and "a", are the most commonly occurring words in a language and are considered to carry little to no information due to their high frequency of appearance (Yao and Ze-wen, 2011). Their presence in the dataset has the potential of adversely affecting the classification results. The most frequent words in the dataset were then identified based on a specified frequency threshold, filtering out all tokens that appeared less than the threshold. This was done to reduce the size of the resultant feature vector and identify the most general set of terms that represents the dataset.

2.1.2 Unigram model with feature reduction through stemming

In addition to the baseline methodology, we applied a technique known as stemming, which reduces words to their basic forms. This process combines words with similar basic forms, for example the words "running" and "ran" are reduced to the base form of "run," thus reducing the overall feature count and increasing the co-occurrence count.

2.1.3 Unigram model with feature enhancement through emoticon retention

One of the disadvantages of removing special characters from the Tweets was that the emoticons, text representations of emotions, were lost. Emoticons are good indicators of expression and emotion, and are frequently used in Tweets. We again performed the steps in the previous methods, but before the removal of special characters from the Tweet, we implemented a series of regular expressions to capture a specific set of emoticons and convert them into unique key words.

2.1.4 Unigram model with word saliency statistics

The saliency, or quality, of the terms in the unigram model were calculated using the Term Frequency - Inverse Document Frequency (TF-IDF) score. The values of the matrix were modified to the TF-IDF score through the equations below.

$$\text{TF(Tweet, term)} = \frac{\text{frequencyOfTerm(Tweet)}}{\text{totalTerms(Tweet)}} \quad (1)$$

$$\text{IDF(term)} = \log \frac{\text{totalDocuments}}{\text{numDocumentsContaining(term)}} \quad (2)$$

$$\text{TF-IDF(Tweet, term)} = \text{TF(Tweet, term)} \cdot \text{IDF(term)} \quad (3)$$

3 Classification

Once the feature vectors were created, the final step of classification was accomplished using supervised machine learning algorithms. In the presented methodology, classification was carried out using single classifiers as well as multiple classifiers. As a reminder, Subtask A is a three class problem, while Subtask B is a two class problem.

3.1 Single Classifier

For this classification method, only a single classifier was used to perform the classification. The feature vector of a Tweet was used as input and the classifier returned the predicted sentiment class for that Tweet.

3.2 Multiple Classifiers

For this classification method, multiple classifiers were utilized to produce the final sentiment class of the Tweet based on a voting system. Each classifier is given a single vote and performs the classification

of the Tweet on its own; casting its vote for which classification should be assigned for that Tweet. The predicted class with the majority of votes is then assigned as the class for that Tweet. We refer to this process as $voting$.

4 Data

The SemEval-2016 (Nakov et al., 2016) training datasets were used for both tasks. The datasets consisted of Tweets with pre-labeled sentiments. Table 1 and Table 2 show the class distributions of the training data.

Table 1: Dataset for Subtask A

# of Tweets	Pos	Neu	Neg
705	345 (48.93%)	164	196

Table 2: Dataset for Subtask B

# of Tweets	Topics	Pos	Neg
3890	59	3215 (82.64%)	675

5 Implementation Specifics

This section discusses the parameters and assumptions made in the implementation of our systems. Our system is freely available for download [1].

5.1 Feature Vector Creation

For the feature vector creation, the stemming process was carried out using the Porter stemmer (Porter, 1997) supplied in the Natural Language Toolkit (NLTK) (Loper and Bird, 2002) platform [2]. The stop word list was manually created and is freely available in our package. The emoticons were retained by converting them to unique tokens using regular expressions. Table 3 shows the emoticons used by the system and their conversion.

For our implementation, we used a frequency threshold of five to filter our features. This parameter was determined during initial development of the system by evaluating several thresholds using 10-fold cross validation over the training data.

5.2 Classifiers

Three classifiers were tested for both subtasks. Subtask A utilized the Naive Bayes Multinomial, Naive Bayes, and J48 decision tree classifiers. Similarly, Subtask B used the Naive Bayes, J48 decision tree, and Support Vector Machine classifiers. These classifiers were used individually as well as with voting. All classifiers were implemented using the open source, freely available Weka (Hall et al., 2009) data mining package [3]. We used Weka's default learning parameters in our experiments.

Table 3: Emoticons

smileEmoticon	:)
frownEmoticon	: (
winkEmoticon	;)
tongueEmoticon	: P
concernEmoticon	: /
grinEmoticon	: D
mirrorGrinEmoticon	D :
winkGrinEmoticon	; D
surpriseEmoticon	: O
tearSmileEmoticon	:')
tearFrownEmoticon	:' (

6 Results

Table 4 shows the overall accuracies acquired by the different classifiers tested for Subtask A. We chose to use a baseline of a unigram model with the frequency threshold set to one. The Naive Bayes Multinomial classifier produced the highest results for Subtask A from the classifiers tested. Further, the enhancements done to the unigram model did not yield a significant increase of accuracy in our tests. The highest accuracy was achieved with the basic unigram model in conjunction with the Naive Bayes Multinomial classifier. With that being said, the basic unigram model with a frequency threshold of five was able to outperform our selected baseline.

Table 5 illustrates the overall accuracies obtained for Subtask B. We swapped the Naive Bayes Multinomial classifier with the Support Vector Machines classifier due to our use of the Tweet's topic as categorical data. The SVM classifier produced the highest overall results. Further, all classifiers except for Naive Bayes performed better than the baseline. Voting did not perform as well with the J48 and SVM algorithms, but still outperformed Naive Bayes. Our highest accuracy was achieved using the unigram model with stemming as features into the SVM classifier.

[1] https://github.com/gerardBriones/twitter-sentiment-analysis

[2] http://www.nltk.org/

[3] http://www.cs.waikato.ac.nz/ml/weka/index.html

Table 4: Overall classification accuracies for Subtask A

	Uni	Uni + Stem	Uni+ Stem + Emot	Uni + Stem + Emot + TF-IDF
NBM	0.577	0.572	0.569	0.557
NB	0.550	0.539	0.540	0.552
J48	0.516	0.549	0.552	0.515
Voting	0.569	0.566	0.562	0.533

Table 5: Overall classification accuracies for Subtask B

	Uni	Uni + Stem	Uni+ Stem + Emot	Uni + Stem + Emot + TF-IDF
NBM	0.692	0.674	0.674	0.612
J48	0.864	0.870	0.870	0.872
SVM	0.879	0.881	0.881	0.870
Voting	0.867	0.865	0.863	0.876

Table 6 shows the results using our unigram, stemming, emoticon retention, and TF-IDF methodology on Subtask A. Our average F1 and average recall scores are higher than the baseline, with our accuracy score having a smaller, but still noticeable increase.

Table 6: Final Evaluation Results for Subtask A

#	System	AvgF1	AvgR	Acc
1	SwissCheese	0.633	0.667	0.646
2	SENSEI-LIF	0.630	0.670	0.617
3	unimelb	0.617	0.641	0.616
4	INESC-ID	0.610	0.663	0.600
5	aueb.twitter.sentiment	0.605	0.644	0.629
31	VCU-TSA	0.372	0.390	0.382
35	baseline (positive)	0.255	0.333	0.342

Table 7 shows the results using our unigram, stemming, emoticon retention, and TF-IDF methodology on Subtask B. Our average F1 and average recall scores are slightly better than the baseline, while our accuracy also slightly decreased.

Table 7: Final Evaluation Results for Subtask B

#	System	AvgF1	AvgR	Acc
1	Tweester	0.797	0.799	0.862
2	LYS	0.791	0.720	0.762
3	thecerealkiller	0.784	0.762	0.823
4	ECNU	0.768	0.770	0.843
5	INSIGHT-1	0.767	0.786	0.864
19	VCU-TSA	0.502	0.448	0.775
20	baseline (positive)	0.500	0.438	0.778

7 Discussion and Future Work

In this paper, we present a method to predict the sentiment of Twitter feeds. We evaluated using a unigram model with three feature modifications: (1) stemming, (2) emoticon retention, and (3) word saliency. These modifications were applied to the unigram model and consumed with machine learning algorithms from the Weka datamining package. The results showed that using a unigram model with a frequency threshold of five in conjunction with the Naive Bayes Multinomial classifiers obtained the highest accuracy for Subtask A, and the unigram model with stemming in combination with the Support Vector Machine classifier achieved the highest accuracy for Subtask B.

Analysis of the results showed that the unstructured nature of word spelling may have played a role in our relatively low accuracies, causing multiple features to be seen as unique, when in actuality they should in fact map to the same feature. We also believe that the mixed results from the inclusion of the TF-IDF score is due in part to the heavily skewed nature of the data. In both Subtask A and Subtask B, the training data was mostly comprised of positively tagged sentiments, overwhelming the other classifications.

In the future, we plan to explore incorporating synonym set evaluations, acronym expansion, and spelling correction to aid in feature reduction. Efforts will also be made to include more contextual information, like sentiment lexicon, and to explore other multiple classifier methods, such as co-training.

References

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Fabrizio Sebastiani. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California, June. Association for Computational Linguistics.

M. F. Porter. 1997. Readings in information retrieval. chapter An Algorithm for Suffix Stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Zhou Yao and Cao Ze-wen. 2011. Research on the construction and filter method of stop-word list in text preprocessing. In *Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on*, volume 1, pages 217–221. IEEE.