"Are You Alright?" Twitter Sentiment Analysis

Mohammad Shahriar Hooshmand
Department of Materials Science and Engineering
The Ohio State University
Columbus, OH
hooshmand.1@osu.edu

Yi Chun Chen
Department of Computer Science and Engineering
The Ohio State University
Columbus, OH
chen.7508@osu.edu

Jerry Ding
Department of Computer Science and Engineering
The Ohio State University
Columbus, OH
ding.667@osu.edu

***Abstract*—This work focuses on the sentiment analysis of tweets posted by a user on Twitter. We implement Naive Bayes classification model, perceptron and support vector machines to identify the sentiment of each tweet. Based on the sentiment analysis performed on the latest 20 tweets of each individual, we finally detemine if the user feels alright or not.**

***Index Terms*—Sentiment, Twitter, SVM, Naive Bayes, Bag of Words**

I. Introduction

Sentiment analysis, also called opinion mining or emotion artificial intelligence (AI), is the process of determining whether a piece of writing is emotionally positive, neutral or negative. This process is commonly used to determine how a person may feel about a particular topic. Sentiment analysis is widely used for a variety of applications.

Artificial intelligence based techniques have been great tools in automating the process of sentiment analysis. With development of new machine learning approaches, many brands and marketers can be able to classify data as positive, negative and neutral for many purposes. In this work, we will use Naive Bayes methods combined with neural network perceptrons and support vector machines (SVM) on the datasets of tweets to determine the sentiment of each tweet and fourthly decide how an individual feels in particular, i.e is she/he alright?

Section II describes the overall problem formulation. Section III-A explains the Naive Bayes categorization method based on the bag of words model. Section III-B introduces the perceptron learning details we implemented in this work. Section III-C focuses on the SVM framework of classification and comparison with the perceptron Naive Bayes results. Section IV describes the statistic results. Finally, in section V we compare our finding with the available studies in the literature.

II. Problem Formulation

Figure 1 shows the framework of each step in this work including preprocessing and data analysis, machine learning analysis and final output derivation. First, we construct a bag-of-words dictionary to analyze the sentiment of a single tweet using the Naive Bayes model. The query variable is the sentiment, represented as a ratio of the conditional probability of positive over negative (sentiment probability), and the "effect" variables are the presence or absence of each word in the tweets. The assumption is that words occur independently in documents, with frequencies determined by the sentiment probability. We construct our dictionary from the dataset available in [2], [3].

Utilizing the bag-of-word dictionary, we dig into the latest 20 tweets of individuals and performed the sentiment analysis on each tweet. The dataset for the sentiment analysis is gathered through the Twitter API (tweepy) [1]. We labeled the sentiment of each tweeter user manually after reading through the latest 20 tweets of them as baselines.

After deriving the sentiment probability associated with 20 latest tweets corresponding to each individual, we use perceptron learning method to calculate weights of each tweet. This mimics the temporal effect of latest 20 tweets which weighs each tweet in the course of time they are posted. Our tweet database includes 130 real Twitter usernames, hand-labeled with the tags positive (1), neutral (0.5) and negative (0). We train the perceptron for 75% of data and test for the rest of 25% unseen data. Then, we use the converged weights to predict each individuals' feeling status, i.e positive (1), neutral (0.5) or negative (0).

In addition, we implement the SVM classification technique to determine the feeling status and compare the results with the Neural Network Perceptron model. The feature space for input data of each individual would be the 20 sentiment probability ratios and the output labels would be positive (1), neutral (0.5) or negative (0).

III. Methods

A. Naive Bayes Categorization

Naive Bayes is a simple conditional probability model to construct classifiers. We treat each word as an instance observation and assume they are conditionally independent,

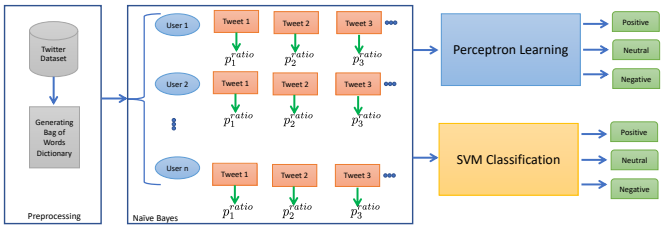

Fig. 1. Framewrok of preprocessing, learning and output performed in this study

and we treat each type of sentiment as a derivation. Denoting the sentiment type as C and the word as x, we generate the Naive Bayes formula as in Figure 2.

Utilizing the bag of words model, we simply calculate the conditional probability of each word counting the occurrence of each word. The prior probability of each sentiment type of tweets is calculated by dividing the number of that type of tweets by the total tweet number. A trick here is that we only consider two types of sentiment: positive and negative. For positive and negative tweets, we count each tweet as 1 occurrence of each type. And for neutral tweets, we count each tweet as 0.5 occurrence of both types. The model generates a ratio of probability of positive sentiment over probability of negative sentiment. Such ratios are used as input for learning methods explained in Sections III-B and III-C.

$$
\begin{aligned}
p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\
&= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k) \cdots \\
&= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,,
\end{aligned}
$$

Fig. 2. Naive Bayes Formula

B. Perceptron Learning

Perceptron learning is a method of supervised learning and linear classifying. It is a simpler version of neural network implementation.

In our implementation, each user in the data set have 20 tweets which are posted in different time intervals. Our model is to give each of the 20 tweets a weight, which is to model how tweets that are posted in different time intervals effect the user's current sentiment state. Intuitively, older tweets should contribute to the user's current emotion less while newer tweets should contribute more. We use the 20 probability values calculated from the Naive Bayes part as input and feeding those to 20 perceptrons. Then we trained the weights using the averaged perceptron algorithm for 100 epochs.

C. SVM

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection [4]. There are several advantages associated to SVMs. They are very effective in high dimensional spaces and in cases where the number of dimensions is greater than the number of samples. SVM also is very memory efficient where a subset of training points are used in decision function (support vectors).

In this part, we use SVM to classify positive, neutral and negative users based on the latest 20 tweets treated as the feature space. SVC class is used to perform multi-class classification on the dataset. SVC within Scikit-learn implements the "one-against-one" approach [5] for multi-class classification. For n number of classes, $\frac{n\times(n-1)}{2}$ classifiers are generated and each will train data from two classes.

As of the kernel function, we use radial basis function (RBF). First, we need to optimize the parameters involved in RBF, namely C and γ. The parameter C trades off misclassification of training examples against simplicity of the decision surface. In the high limits of C, all training examples are enforced to be classified correctly. γ parameter indicates how significant a single training example would be in the model. The larger γ is, the closer other examples must to be affected. We use "GridSearchCV" routine within Scikit learn package to perform these parameters tuning on 25% of training data chosen randomly as validation set.

IV. Results

A. Naive Bayes Categorization

Using the dataset from Kaggle Sentiment Competition consisting over 100 million tweets [3], we construct a dictionary of prior and conditional probabilities for words. Training and testing with the dataset using 90% training and 10% testing cross-validation, we only achieve an accuracy at 36.67% for a single tweet sentiment analysis.

B. Perceptron Learning

We train our model using the dataset mentioned above [3] and test it on arbitrary users we found online, each with their latest 20 tweets. The result is as Figure 3:

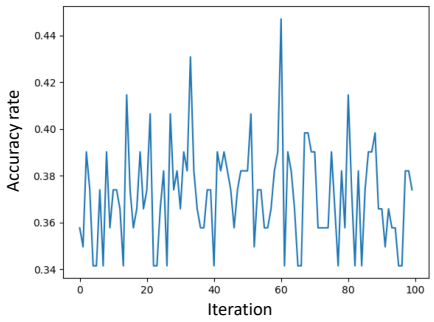

Fig. 3. Accuracy rate in 100 epochs for perceptron learning

The perceptron learning method gets about 37.0% accuracy. We can improve this result by having a more consistent labeling on the arbitrary users tweets.

TABLE I
CLASSIFICATION METRICS REPORT FOR SVM

Class Label	Precision	Recall	f1-score	Support
Negative (0)	0.36	0.44	0.4	9
Neutral (0.5)	0.29	0.15	0.2	13
Positive (1)	0.31	0.44	0.36	9
Micro avg	0.32	0.32	0.32	31
Macro avg	0.32	0.35	0.32	31
Weighted avg	0.31	0.32	0.31	31

C. SVM

We train our SVC model on the 75% of all data chosen randomly as the training set and tested the performance on the rest of unobserved 25% data. Table I shows the performance, accuracy and the scores corresponding to this model.

Using the approach explained above, we got $C = 0.0625$ as the best hyperparameter using "squared-hinge" loss. With these settings, the overall accuracy from SVM model is 32.3%.

V. COMPARISON WITH THE AVAILABLE DATA

There are various studies reported in the literature on sentiment analysis of texts. Specifically, a work by Agarwal *et.al* [6] focused on the sentiment analysis on Twitter data. They performed two classification tasks: 1) positive versus negative and 2) positive versus negative versus neutral. The latter analysis is pretty similar to the task we followed in this project. The set-up for this analysis is reported to be SVM where they tune the C parameter using embedded 5-fold cross-validation. 1709 instances for each class has been chosen. Using unigrams baseline feature method, they achieved 56.58% accuracy. The F1 measure has been reported as 58.86,56.58 and 56.20 for positive, neutral and negative sentiments, respectively. Referring back to our finding using SVM in I, the accuracy of this work is around 23% better than our model. One important factor which plays role in this difference is the size of the dataset we used which is around 10% of work by Agarwal *et.al*. Also, they have implemented the unigram model on the features which increases the gain by 23.35% over chance baseline. We expect that such type of feature analysis can boost the performance significantly.

VI. CONCLUDING REMARKS

In this study, we perform sentiment analysis on the latest 20 tweets of a dataset of Twitter users to predict if the user feels perfect, neutral or bad. We use Naive Bayes model to carry out the sentiment analysis on each tweet (using bag of words model) and then we calculate the conditional probability ratios. Using two learning approaches, namely perceptron and SVM, we build up a framework to predict whether a given user is alright or not. The accuracy of both approaches lie within a same range around $32 - 37\%$. This accuracy can be increased by using a more robust word filtering instead of bag of words, larger dataset of users and implementing temporal models to link the time-dependencies of each tweet before feeding in the perceptron and SVM models.

REFERENCES

[1] J. Roesslein, "Tweepy documentation" Online at http://tweepy. readthedocs. io/en/v3 5, 2009.

[2] N. Sanders, "Twitter Sentiment Corpus".

[3] University of Michigan "Kaggle Sentiment Competition, online at http://thinknook.com/wp-content/uploads/2012/09/Sentiment-Analysis-Dataset.zip".

[4] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of machine learning research 12.Oct (2011): 2825-2830.

[5] Knerr, Stefan, Léon Personnaz, and Gérard Dreyfus. "Single-layer learning revisited: a stepwise procedure for building and training a neural network." Neurocomputing. Springer, Berlin, Heidelberg, 1990. 41-50.

[6] Agarwal, Apoorv, et al. "Sentiment analysis of twitter data." Proceedings of the Workshop on Language in Social Media (LSM 2011). 2011.