Sentiment analysis on Twitter data using semeval 2016 data

Zepeng Huo
Department of Computer
Science and Engineering
Texas A&M University
Email: Guangzhou92@tamu.edu

Pradnyil Sranjay
Department of Industrial
and Systems Engineering
Texas A&M University
Email: pradnyil@tamu.edu

***Abstract*—Sentiment analysis is getting popular on social media. In this paper, we propose a method to classify the emotion valence of a given tweet. The whole framework can be separated as two parts: first, we cluster the similar tweets based on the fact that all data of semeval-2016 are collected from specific topics. And the topic can be distinguished as three different parts. Based on the result of clustering, we feed the tweet into the Tensorflow network to classify.**

***Keywords*—Social Media, Sentiment Analysis, Clustering, Neural Network, Tensorflow.**

I. Task and Problem

In Semeval 2016 competition task 4, Twitter sentiment analysis, we need to find out a way to classify sentiment of tweets. The data they provided is of two parts, training set and testing set. Each dataset consists of a couple thousand tweets and each tweets are labeled as either 'positive' or 'negative'. There are 5 subtasks and we only focus on the first one, binary classification. But by looking at the data from other subtasks which give one more label for each tweet, called 'topic'. Even though subtask A didn't have the label, the tweets are all the same. Thus, in order to make use of topic relatedness, we cluster the tweets in three broad categories. In each cluster, we have a particular classification model to classify that category. As for classification model, we use the Tensorflow neural network. The reason we want to use this model is because it's robust to noisy input but also sensitive to distinguishable text which can carry emotional expression.

II. High-level description of approach

Before the classification, we need to tokenize all the tweets in Twitter context. This step is necessary because some expression may be unique on social media like Twitter and thus carry specific emotion. For example, the emoticon ':)' means happy or smiley, and it is highly likely related to positive sentiment. But if we use normal tokenization method, it may separate it as ':' and ')', so it loses its original meaning. The tokenization method we use is twokenize. It's specific to Twitter and make every token intact, without adding or losing meanings. After tokenization, we can feed each tweets into a binary classification. Since we use two steps to classify each tweet, we'll introduce the problems we tackled on each step.

As for clustering, we want to make use of the hidden information under the data provided by Semeval-2106. Even though this task doesn't require to classify topics, we still want to have related tweets to be classified when they share similar topics. At the original data in other subtasks, the dataset has 60 topics. But most of them are highly related, for example, 'apple' and 'apple watch', 'Donald Trump' and 'Hillary'. Usually, when people are talking about this similar topics, they tend to use similar words carrying similar emotions. But when they talk about other topics, this words can express opposite sentiment, thus making the classification give unsatisfactory results. So we want to gather similar tweets together when the topic label is hidden in subtask A. The reason why we choose three categories is that we found out the apparent difference among all the tweets and each tweet will definitely fall into one of three categories, which are brand/product, person/people, organization/association. The number of categories is not too big, such as originally 60, to make the model underfitted. Nor it's not too small to make the less related tweets have fallen into same cluster.

III. Specific System Implementations

In the part of clustering, we have our novelty in the system. First, as for converting the tweet into a mathematical representation, most common way is to use bag of words and word embedding. Then the whole tweet is concatenation or averaged vector of word embedding. These methods have problems of either making tweets vectors too long or losing lots of latent information by averaging word vectors. In our system, we use Doc2Vec. It can learn the vector of one sentence of one document by going through the corpus we provide. It can preserve the original information in the tweet at the greatest extend it can and also without increasing the tweet vector dimension unnecessarily. Both the time complexity and space complexity can come to a balance.

Since the dataset Semeval provides is relatively small (up to 6000 tweets for training), we may have the problem of overfitting the data thus making the noisy input contaminate the model. If some of the tweets are using sarcasm and making positive adjectives carrying negative emotions. If we only train on these tweets, it's highly likely that similar positive words will be classified negative in other tweets, even though these tweets didn't use sarcasm. In the light of this, we propose a method to make the vector initialization closer to a well-established understanding of English words in Twitter. By initializing the word embedding using pre-trained word vectors, the document vector is more likely to come to a optimum. For the pre-trained word embedding, we used Glove, Global Vectors for Word Representation. The pre-trained word

vectors are trained on 2 billion tweets, ending having 27 billion tokens and 1.2 million vocabulary. After feeding pre-trained word embedding into the Doc2Vec model, we make our model carry more well-known usage of English words in Twitter.

After converting each tweet into a vector, we start clustering the tweets into three categories we discussed before. The cluster method we used is k-means. It can cluster most similar vectors together and make different vectors far away from each other. Based on the dimensionality of the vector space, 300, k-means performs well in time complexity.

IV. Evaluation, Metric and Experimental results

V. Conclusion

Appendix A
Some add-ons

Acknowledgment

The authors would like to thank...

References

[1]