EECS 445: Progress Report

Nand Dalal
University of Michigan
nndalal@umich.edu

Charles Lewis
University of Michigan
noodle@umich.edu

Jeffrey Lin
University of Michigan
yulong@umich.edu

Eric Taseki
University of Michigan
taseskie@umich.edu

Bo Chen
University of Michigan
boch@umich.edu

Abstract

Our team has chosen to solve the 'Partly Sunny with a Chance of Hashtags' challenge hosted by Kaggle. The goal of the challenge is to provide accurate prediction for twenty-four labels associated with tweets related to the weather. According to Kaggle, "The challenge is to analyze the tweet and determine whether it has a positive, negative, or neutral sentiment, whether the weather occurred in the past, present, or future, and what sort of weather the tweet references." Upon successful completion of this project, we will have developed models to approach a large portion of social media data. Thus far, we have been able to tokenize the tweets by applying a stemmer and spell checker. We have also been able to pass this tokenized data to a logistic regression model to predict whether a particular tweet is referencing 'cloudy' weather or not. Preprocessing the data has proven to be the toughest and most important aspect of the challenge. Moving forward, we will be apply more algorithms to tokenize the data such as PCA and sparse coding to reduce the data's dimensionality. Also, we will be applying multilayer perceptrons and GMM in order to predict multiple labels and discover complex relationships between the input features and output.

1 Introduction

The timeframe and subject matter of the 'Partly Sunny with a Chance of Hashtags' challenge hosted by Kaggle makes it an attractive competition for our team to take part in. The problem is to be solved using the analytics of nearly 80,000 tweets for the prediction of a confidence score for twenty-four labels indicating various sentiments, time, and the weather categories. Our provided training data by CrowdFlower Open Data is an attempt to facilitate the use of large datasets through the creation of labels paired with each data entry for machine learning applications. Through competing in this challenge, we are hoping to generate methods for accurate prediction of scores for data labels derived from a large data set and generalize these machine learning algorithms for other such potential applications. Upon successful completion of this challenge, not only would we have developed algorithms for accurately predicting the sentiment, time, and weather condition given a tweet, but more imporantly we would have achieved the validation of such data organization approach, which can be applied to existing datasets for efficient machine learning applications.

2 Proposed Method

In order to understand the methods being used to solve this classification problem, we will start by examining the data provided by Kaggle.

Kaggle has provided both training and testing data. The testing data has not been labled, and will be used for ranking in the competition. The training data consists of three major items: the raw tweet, locations, and a confidence score corresponding to each of twenty-four labels. The twenty-four labels are further split into three categories: sentiment, when, and kind. The people who rated these tweets were allowed to pick only one label for the sentiment and when categories, but were allowed multiple selections for the kind category. Having a single person rating each tweet introduces a bias, so each tweet has been reviewed by multiple people, who may not agree on the assigned labels for a particular tweet. As a result, the sentiment and when categories' sum of labels will usually be close to one each, however this is not considered in the code since the proposed algorithm should learn this and account for it in the result.

Now that we understand the organization of the data, we can proceed with the proposed method for processing the input.

Feature extraction from the raw tweets is arguably the most imporant part of the project. Upon inspection of the training data, we realized that words in the tweets are often spelt with the intention of a four year old. Also, punctuation and other characters in the tweets often times do not affect the tweet's classification. Therefore, we have started the preprocessing stage by tokenizing each tweet. This involves two major steps. First, we use the NLTK library's Lancaster Stemmer to stem each word. Second, we use Peter Norvig's toy spell checker to check if the word is spelled correctly. However we will attempt to implement a spell corrector, since throwing out all mis-spelt words decreases the potential power given by the data. These two steps in the preprocess have proved to reduce the number of features from a few hundred thousand to just below 7,000.

Glancing over the training data, a person is clearly able to discern words in common between different tweets that affect the labeling in a similar fashion. This can be achieved by implementing PCA to further reduce the data's dimensionality followed by sparse coding to learn bases from the raw tweets. This will allow the features to not only be distinct words but rather meaningful phrases and groups of words. This takes care of the input data.

Whilst attempting to reduce the data's dimensionality, we will be feeding the resultant data matrix to multiple machine learning algorithms. We will be using the deep learning algorithms already implemented in the Theano library. The Theano tutorials demonstrate the use of logistic regression and a multilayer perceptron. Even though logistic regression does not make too much sense given multiple labels that require classification, we will be using it to familiarize ourselves with the Theano library as well as NumPy, the library for N-dimensional array objects that Theano uses. The multilayer perceptron will be our primary algorithm. Specifically, the input layer will be a set of nodes corresponding to the number of occurences of a certain feature in the training example, and the output layer will be a set of nodes corresponding to the twenty-four labels. In addition, we will attempt to apply GMM to the data to classify what kind of weather the tweets correspond to, as they might be associated with multiple kinds of weather.

3 Related Work

There are many different models used in extrapolating sentiment data from tweets. The most popular ones include Naive Bayes classifier, Maximum Entropy, and SVM. The most popular algorithm happens to be Naive Bayes due to it is simplicity and ease of implementation. The results of a simple algorithm on a very large data set has been shown to produce better results than a more complicated algorithm (Lin_Kolcz). Processing of the data is a very key factor in the sentiment analysis. There are a multitude of ways to process and tokenize each tweet. Features can be individual words or sequence of n words.

Unigrams, the "...easiest and most used approach (Pang et al.) reported an accuracy of 81.0%, 80.4%, and 82.9% for Naive Bayes, MaxEnt and SVM respectively in the movie-review domain. This was found to be closely similar to accuracies obtained in twitter classification which were 81.3%, 80.5%, and 82.2% respectively."

The combination of unigrams and bigrams in the feature set showed an improvement with the Naive Bayes and MaxEnt results but a decline for SVM.

Thus far, we have focused a lot on the processing the tokens. With various means of processing tokens we have between 6,000 and 60,000 unique tokens. The data analysis performed in other more in-depth research was on data sets much larger than ours. Also, we are facing tighter memory constraints than larger research projects as some of them are implemented on large data systems such as Hadoop. Therefore, we must keep our memory use relatively low which causes issues with speed.

4 Experimental Results

As previously metioned in the section pertaining to the proposed method we have implemented a first iteration of tokenizing our input data. This involved applying a word stemmer from the NLTK library and a spell checker developed by Peter Norvig. A combination of these two methods reduced the token count from a raw 80,000 spanning the entire training set to just over 6,000. However, we must keep in mind that this method also involves throwing away mis-spelt words as well as getting rid of punctuation including emoticons, which may affect the sentiment analysis. After preprocessing the data, we have plugged our data into Theano's logistic regression tutorial in order to familiarize ourselves with the Theano library. We chose to only learn about 'cloudy' tweets, so we used 500 tweets and the corresponding 'cloudy' weather label to predict 'cloudy'-ness in another set of 500 tweets from the training examples. This is because the test data provided by Kaggle has not been classified. Theano's logistic regression model correctly labeled 480 out of 500 of the test examples from the training data, achieving 96% accuracy.

5 Future Milestones

While performing the first iteration of preprocessing the training data, we found that tokenizing the tweets contained in our 17MB file translated to a file above 4GB in size. Loading this CSV file on our personal computers eats up the available RAM and provides for slow computation. Therefore, moving forward we will have to look better computing options.

As mentioned above, the preprocessing stage of this challenge may prove to be the hardest and most crucial. Moving forward, we will be implementing PCA to reduce the dimensionality our data. Sparse coding will also help in reducing the number of features by learning bases from the raw tweets.

Since we gained experience with the Theano library, specifically the logistic regression model, we can move forward with the machine learning algorithms by implementing a multilayer perceptron with inputs being the occurences of the features and outputs being the category labels. We will also experiment with multiple logistic regression models to learn the kind of weather associated with each tweet, since logistic regression proved to yield a high accuracy.

Given the fact that we are experimenting with multiple models including logistic regression, multilayer perceptron, and GMM, we will have to perform cross-validation using each model. We will employ the K-fold cross-validation method to divide our data amongst our various models.

In a timeline view, we will continue working on feature/dimensionality reduction. In parallel, we will be feeding our resulting input data to multiple models.

6 Conclusion

From this challenge, we want to demonstrate that it is possible to classify sentiment, time, and weather conditions with a limited amount of words. Being able to quickly analyze short sentences and small phrases is crucial as the general population is slowly receiving news and information from social medias like Twitter, Facebook, Google+, and etc. The results that we have achieved from Theano's logistic regression thus far is very hopeful. Currently, we have performed a first iteration of tokenizing the raw tweets. We have also been able to apply a machine learning algorithm, namely logistic regression to classify one attribute of the tweets. Moving forward, we wish to analyze the full data set provided by Kaggle. This will entail further tokenizing tricks including PCA and sparse coding as well as the application of more complex algorithms including multilayer perceptrons and GMM.

References

[1]Kaggle, Partly Sunny with a Chance of Hashtags, `http://www.kaggle.com/c/crowdflower-weather-twitter`

[2]Theano Deep Learning, `http://deeplearning.net/tutorial/`

[3]NLTK Lancaster Stemmer, `http://nltk.org/api/nltk.stem.html`

[4]Peter Norvig, *How to Write a Spelling Corrector*, `http://norvig.com/spell-correct.html`

[5]Hannak et al., *Measuring and predicting sentiment on Twitter*, `http://www.janysanalytics.com/ussi/pdf/Measuring-and-predicting-sentiment-on-Twitter.pdf`

[6]Lei Zhang et al., *Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis*

[7]Jimmy Lin & Alek Kolcz, *Large-Scale Machine Learning at Twitter*

[8]Anthony K Jose et al., *Twitter Sentiment Analysis*