Real Time Twitter Sentiment and Opinion Analysis

Purvasha Das, Bowen Shen, Yitao Wu, *University of Southern California*

Abstract

Sentiment analysis, also called opinion mining, has become a highly popular research area both in business and daily life with the increasing development of social media. Generally speaking, sentiment analysis is the process of defining and categorizing opinions in a given piece of text as different emotion attitudes, that is positive, negative, or neutral. Twitter, as a popular social media tool, allows people to express their opinions on a spectrum of topics with some short but meaningful tweets, which provide perfect data for sentiment analysis. In this paper, we first formulate our project problem and explain theories of all the tools we will use. Then, we implement and examine two different machine learning methods (Naive Bayes, and Long Short Term Memory (LSTM)) dealing with twitter sentiment analysis, and compare the results gathered by the two methods. Finally, we perform sentiment analysis using one machine learning method on the tweets generated by tweepy API, visualize, and analyze the results.

Keyword: Sentiment Snalysis, Twitter, Naive Bayes, LSTM

I. INTRODUCTION

It's extremely intriguing to find the real sentiments behind what exactly are people thinking. This vastly growing availability and popularity of opinion-rich resources such as online review sites and personal blogs, new opportunities and even challenges arise as people now can actively use information technologies to seek out and understand the opinions of others. That's what makes sentiment analysis such an expansive and interesting field. Sentiment analysis–also called opinion mining–is the process of defining and categorizing opinions in a given piece of text as positive, negative, or neutral. With technology's increasing capabilities, sentiment analysis is becoming a more utilized tool for businesses and any domain. The performance of any organization depends on its customer and hence their sentiments. Social media monitoring tools use it to give their users insights about how the public feels in regard to their business, products, or topics of interest. The sudden explosion of activity in the area of opinion mining and sentiment analysis, dealing with the computational treatment of opinion, sentiment, and subjectivity in text, has thus occurred at least in part as a direct response to the surge of interest in organizations or systems that directly deal with opinions as a first-class object. Sentiment Analysis became a huge news when it was discovered that Facebook used emotional contagion to decisively influence its user's emotional output by flooding its newsfeed with positive or negative posts. [1]

Twitter too being a popular microblogging service allows its users to create status messages called tweets. These tweets express opinions (hence providing data for analysis) on a spectrum of topics. Twitter has lately been a media to get dataset for conducting sentiment analysis that could be useful for consumers indulging in research on a particular product or service or even the marketers researching public opinion about their product.

With each one having its own set of advantages and disadvantages, we opted for Naive Bayes algorithm which we have explained in later part of the report. We have also explained our use of python because of it's both compatibility and popularity in the Natural Language Processing area in the Experimental Setting section.

We find that consumption of goods and services is not the only motivation behind people's seeking out or expressing opinions online. A need for political information is another important factor. The recent scenario of user's hunger for a reliance on online advices, opinions and recommendations is the most distinct and compelling reason behind the sudden "surge" of interest in new systems that deal directly with opinions as first class objects. And this propelled our project to construct a real time twitter sentiment analysis. [2]

The project objective is to conduct a real time twitter sentiment analysis. Our project involves the use of social networks, data analytics, cloud, and machine learning. Sentiment analysis refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine and analyze the information received.

Collectively, the aim is to analyze the emotion state of twitter handlers, evaluate users' attitude

2. THEORIES

2.1 Long Short Term Memories

2.1.1 Deep learning methods in Natural language processing task.

Natural language processing is all about creating systems that process or understand languages to perform certain tasks, including Question Answering (Siri, Alexa, Cortana), Machine Translation (Google Translator), Speech Recognition (understanding human speaking), Image-Text transfer (generate image based on text, and vice versa), and Sentiment analysis (judge emotion tone given an input text). In the past, NLP usually requires considerable amount of domain knowledge in linguistics (like phonemes or morphemes). In the past few years, deep learning [3] has shown its incredible power in image understanding (computer vision) without specific domain knowledge. Provided enough data as well as compute power (GPU, specifically), deep learning can extract huge amounts of characteristics from images and classify or even generate some images. Similar to images, languages also contain numerous information or details that can't be noticed by humans, which enables deep learning as a possible and efficient solution to most NLP tasks, including sentiment analysis.

2.1.2 Word2Vec

If we directly put the sentence as the input of the deep learning model, then training will become impossible, as there is no way for us to compute dot-product of matrix or backpropagation for every single sentence, which is the core computation in deep learning.

Figure 1. Sentiment analysis without preprocessing.

To solve this problem, we need to transfer those sentences into vectors, which computable in the deep neural network. The similar vectors should share some similar characteristics (like and enjoy), and totally different vectors should have opposite characteristics.

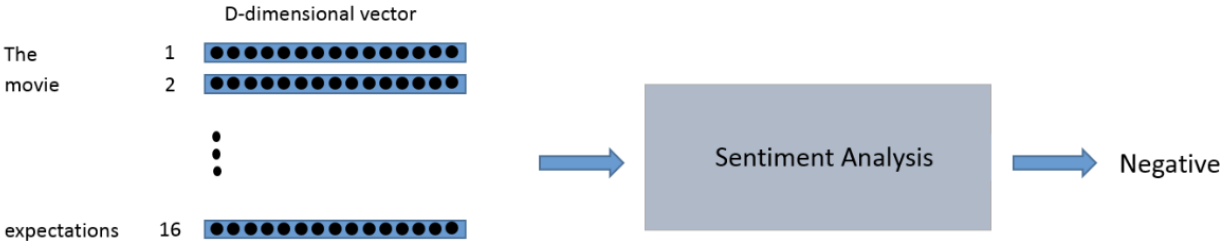

Figure 2. Sentiment analysis with preprocessing.

One of models to create word vectors is called “Word2Vec”. The Word2Vec model is trained by taking each sentence in the dataset, sliding a window of fixed size over it, and trying to predict the center word of the window, given the other words. Using a loss function and optimization procedure, the model generates vectors for each unique word [4].

2.1.3 Recurrent Neural Networks (RNNs) [5]

Normal neural network (like CNN) concentrates on the current information, like specific images. In this case, training progress won’t be influenced by the previous training, which makes sense for image classification or other images, considering images are independent with each other. However, for sequential information like words in the sentence, every element may influence each other and previous words are needed when predicting the following words. To solve sequential issue, “memory” of previous results calculated is needed when training current element. And RNN, with such kind of “memory”, recurrently performs the same task for every element in a sequence.

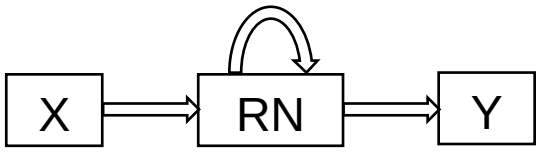

Figure 3. RNN high-level model.

The diagram (Fig. 3) above shows RNN can recurrently generate input for next step as well as output y, which is illustrated by the right recurrence formula. Following three equations explain calculation of RNN. For (1),

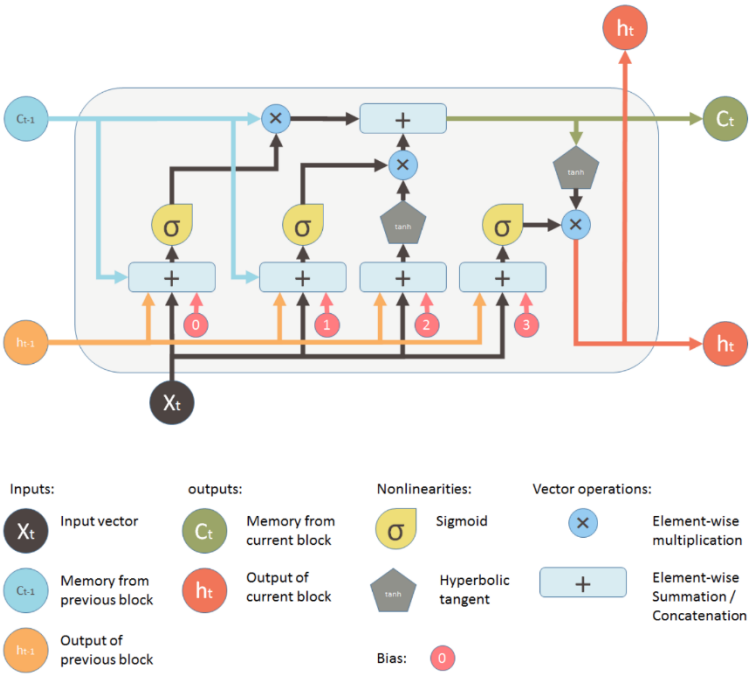

Figure 4. Illustrative expression of LSTM inside computation [7].

ht is the hidden states at the time step t (result of RNN), ht-1 is hidden states storing previous calculation at the time step t-1, xt is the input at the time step t. fW is the RNN function defined in (2) where Whh is the weights for ht-1, and Wxh is the weights for xt. For (3), the output at the time step t (yt) can be calculated by the bottom formula. Although RNN usually works well with short texts, it often fails when dealing with longer text or time steps because of vanishing and exploding gradients.

$$h_t = f_W(h_{t-1} + x_t). \tag{1}$$

$$h_t = tanh(W_{hh}h_{t-1} + W_{xh}x_t). \tag{2}$$

$$y_t = W_{hy}h_t. \tag{3}$$

2.1.4 Long Short Term Memory Units (LSTMs) [6]

LSTMs can solve the problem of RNN by replacing the simple update rule of vanilla RNN with a gating mechanisms. At a high level, they make sure that the hidden state vector can encapsulate information about long term dependencies in the text.

In LSTM, we first compute an activation vector as (4). We then divide a into four vectors ai, af, ao, ag where ai consists of the first H elements of a, af is the next H elements of a, etc. We then compute the input gate g, forget gate f, block input g and output gate o as (5), where σ is the sigmoid function. And finally, we get the next cell state Ct and next hidden states ht as (6), where × is element-wise multiplication. Fig. 4 shows the whole process.

$$\mathrm{a} = W_x x_t + W_{hh}h_{t-1} + \mathrm{b}. \tag{4}$$

$$\mathrm{f} = \sigma(a_f), \mathrm{i} = \sigma(a_i), \mathrm{g} = \tanh(a_g), \ \mathrm{o} = \sigma(a_o) \tag{5}$$

$$C_t = \mathrm{f} \times C_{t-1} + \mathrm{i} \times \mathrm{g}, \ h_t = \mathrm{o} \times \tanh(C_t) \tag{6}$$

2.2 TextBlob

As stated earlier just like images, languages also contain numerous information or details that go unnoticed by humans enabling deep learning as a possible and efficient solution to most NLP tasks, including sentiment analysis. Our second choice was TextBlob. It is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

Programming as we know, can be distinguished into the two complementary tasks of description and proceduralization. Among the various paradigms for computer programming – such as logical, declarative, procedural, functional, object-oriented, and agent-oriented – the object-oriented and agent-oriented formats most closely embody human storytelling intuition. Textblob stands on NLTK and pattern. [8]

NLTK (Natural language ToolKit) is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

On the other hand the pattern.en module, contains a fast part-of-speech tagger for English (identifying nouns, adjectives, verbs, etc. in a sentence), sentiment analysis, tools for English verb conjugation and noun singularization and pluralization, and a WordNet interface.

Since our analysis is concentrated on sentiment analysis, we will focus on that aspect of both pattern and NLTK. [9]

Written text can be broadly categorized into two types: facts and opinions. Opinions carry people's sentiments, appraisals and feelings toward the world. The pattern.en module bundles a lexicon of adjectives (e.g., *good*, *bad*, *amazing*, *irritating*, ...) that occur frequently in product reviews, annotated with scores for sentiment polarity (positive ↔ negative) and subjectivity (objective ↔ subjective). The sentiment() function returns a (polarity, subjectivity)-tuple for the given sentence, based on the adjectives it contains, where polarity is a value between -1.0 and +1.0 and subjectivity between 0.0 and 1.0. The sentence can be a string, Text, Sentence, Chunk, Word or a Synset.

The positive() function returns True if the given sentence polarity is above the threshold. The threshold can be lowered or raised, but overall +0.1 gives the best results for product reviews. Accuracy is about 75% for movie reviews.

As for NLTK, we were interested in using its sentiment analyzer tool. A Sentiment Analyzer is a tool to implement and facilitate Sentiment Analysis tasks using NLTK features and classifiers, especially for teaching and demonstrative purposes. Its sentiment analysis tool is based on machine learning approaches.

Thus the Textblob also has the following features:

- Noun phrase extraction
- Part-of-speech tagging
- Sentiment analysis
- Classification (Naive Bayes, Decision Tree)
- Language translation and detection powered by Google Translate

- Tokenization (splitting text into words and sentences)
- Word and phrase frequencies
- Parsing
- n-grams
- Word inflection (pluralization and singularization) and lemmatization
- Spelling correction
- Add new models or languages through extensions
- WordNet integration

TextBlob aims to provide access to common text-processing operations through a familiar interface. The sentiment property returns a namedtuple of the form Sentiment(polarity, subjectivity). The polarity score is a float within the range [-1.0, 1.0]. The subjectivity is a float within the range [0.0, 1.0] where 0.0 is very objective and 1.0 is very subjective.

Bayes theorem, gives us a clearer view regarding a sentiment analyzer.

The Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features. Given a class variable and a dependent feature vector through , Bayes' theorem states the following relationship:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots x_n \mid y)}{P(x_1, \dots, x_n)} \tag{7}$$

Using the naive independence assumption that

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y), \tag{8}$$

Thus (7) can be simplified to:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)} \tag{9}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y), \tag{10}$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$ from (10) ; the former is then the relative frequency of class *y* in the training set. The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i \mid y)$. [21]

In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering.

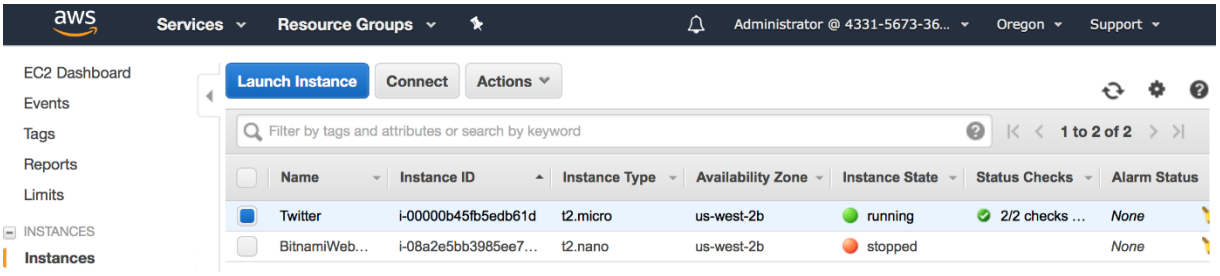

Figure 5. Screenshot for aws EC2 instance.

While training the data, what we wish to determine is if a datarow should be classified as positive or negative. The above calculations shows that we have to calculate the probabilities of each classification and probabilities of each feature falling into the classification. The easiest to generate features is to split the text into words, word in a review will then be a feature that we can then work with. In order to do this, we'll split the reviews based on whitespace. The words could be counted each time a feature occurs in a positive review, or a negative review or in our case of interest, even a neutral review. The word counts can be converted into probabilities and multiply them to get the predicted classification.

Any model requires a training data to estimate the necessary parameters. Getting good results on training set could mean that the model is overfit and thus is just picking random noises. Thus, testing on a dataset that wasn't used to train the model is the best way to review any model for its performance. [10]

2.3 Performance comparison between two algorithms

Finally, we compare LSTM RNN with Naive Bayes method in three different machine learning performance metrics: accuracy, training time, linearity, as shown in Table I.

TABLE I
Performance Comparison Between LSTM and Naive Bayes

Performance Metrics	LSTM RNN	Naive Bayes
Accuracy	Relatively high (80 – 90%)	Relatively low (around 70%)
Training time	Very long (LSTM contains four gates which increase training parameters exponentially)	Short (both training and testing linear time complexity, highly time efficient)
Linearity	High complexity (quite a number of matrix multiplication, high cost)	Less complexity (almost linear for both training and testing, easy to execute)

We can find that although LSTM provides high accuracy, Naive Bayes would be a better choice for general applications due to the extremely high complexity and long training time.

3. EXPERIMENTS

We first train our model using LSTM RNN. Deep learning model is famous for hard-training due to huge amounts of training parameters. To alleviate local machine burden, we decide to run our code on aws EC2 instance [11]. Amazon EC2 provides various instance types to enhance our application performance.

We need a well-round instance built with all the software environment, so we choose Bitfusion Ubuntu 14 TensorFlow [12] on AWS Marketplace.

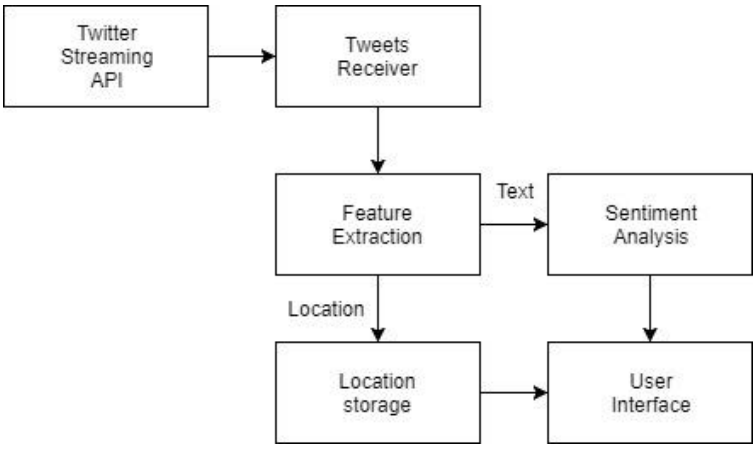

Figure 9. Block Diagram for our program.

For software environment, Keras [13] provides all we need for training a LSTM twitter sentiment analysis. Keras is an open source neural network library written in python, running on the top of Tensorflow or Theano, and wildly used by deep learners all over the world. For dataset, we choose Twitter Sentiment Analysis Training Corpus (Dataset) [14].This dataset contains 1,578,627 classified tweets. Each row is marked as 1 for positive sentiment and 0 for negative sentiment. Besides, Global Vectors for Word Representation (GloVe) [15] is an unsupervised learning algorithm for obtaining vector representations for words, working as Word2Vec tool mentioned before. Jupyter notebook [16] is perfect tool for writing python code, as we can get real time output after a cell of code.

The second part is to write code for twitter stream listener and user interface. We have many choices when it comes to implement our own project. We narrowed it down to use python, concerning the compatibility and its popularity in the Natural Language Processing area. The purpose of our project, is to use social media feeds as the source, after processing the text, accurately present how people from different countries feel about certain user specified keywords. The basic idea is illustrated in the following block diagram.

Our very first step is to get the live feed of a selected social media. Twitter is one of the most popular social media network out there. From 2016 twitter user statistics [17], there are over 310 Million monthly active users, and 29.2% of total US social media users are Twitter users. In fact, twitter is also one of the most used social media for companies, 65.8% of US companies with 100+ employees use twitter for marketing. No wonder all the data scientist from different companies are working closely with twitter data to analyze and predict user behavior. There are 2.5x increase for customer service conversation that was held on Twitter over the past two years. By choosing Twitter as our main social media source, we are ready to transform our project to fit in real world data analysis.

In additional, twitter Apps are well developed and easy to use. We can create a twitter application at twitter developer website. By doing this, we can obtain our unique consumer keys and access tokens for authentication. Those keys will grant us the access to real time social media feeds (real time data sets) from twitter. All the data that provided by twitter API will be using JavaScript Object Notation (JSON). JSON is based on key-value pairs, there will be attributes that associated with values [18]. A typical twitter JSON will be like (randomly selected user tweet that fed to our terminal):

{"created_at":"Thu Nov 16 00:13:26 +0000 2017","id":930951911439728600,"id_str":"930951911439728640","text":"Emergency vehicles. right shoulder in #OverlandPark on 69 Hwy SB after 119th St #KCtraffic https://t.co/uDvucNmUVj","source":"TTN KC traffic","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in_reply_to_user_id":null,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"user":{"id":249823200,"id_str":"249823200","name":"TTN Kansas City","screen_name":"TotalTrafficKC","location":"Kansas City, MO","url":null,"description":null,"translator_type":"none","protected":false,"verified":false,"followers_count":1904,"friends_count":99,"listed_count":92,"favourites_count":4,"statuses_count":100333,"created_at":"Wed Feb 09 21:20:32 +0000 2011","utc_offset":-21600,"time_zone":"Central Time (US & Canada)","geo_enabled":true,"lang":"en","contributors_enabled":false,"is_translator":false}

In this project, we are interested in the user's' tweet sentiment (Positive, Neutral, Negative) about selected keywords and where are those users from.

It will be an exhausting project if we were going to start everything from scratch. Thankfully, there are lots of well-developed packages to handle different tasks. For this project, we integrated tweepy, textblob, and matplotlib.

According to tweepy documentation [19], tweepy is an open-sourced github project that enables python to communication between twitter platform and use their API. In the project, we imported three libraries from tweepy, which are StreamListener, OAuthHandler, and Stream. After providing the correct consumer keys and access tokens we obtained from creating twitter application, we are ready for retrieving real time tweets. It worth a mention that stream.filter is provided, that allows us to select keywords so that we are only receiving relevant tweets. That saved us a lot of trouble for information extraction, also computing power. TextBlob was introduced, we integrated this library to implement our sentiment analysis, which means we will extract the attribute “text” from twitter JSON to perform the analysis.

Lastly, we integrated matplotlib into this project [20]. Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. For the purpose of intuitively display how people feels about certain keywords, we created pie chart of sentiment analysis. We also included the country name where the twitter was sent. The country code information is stored in a dictionary called default_data. It stores the location attribute “country_code” as key, and the amount of its appearance as value. Every time a new tweet’s location has been extracted, we search it in the dictionary, update the count of it or add it as a new key, based on the fact that if the “country_code” is already stored. Everything displayed for our users will be in real time, as the result figure updates itself while relevant tweets keep coming in.
Also, every time we run the program, received twitter JSON will be saved in AWS S3 bucket through aws kinesis firehose.

4. RESULTS AND ANALYSIS

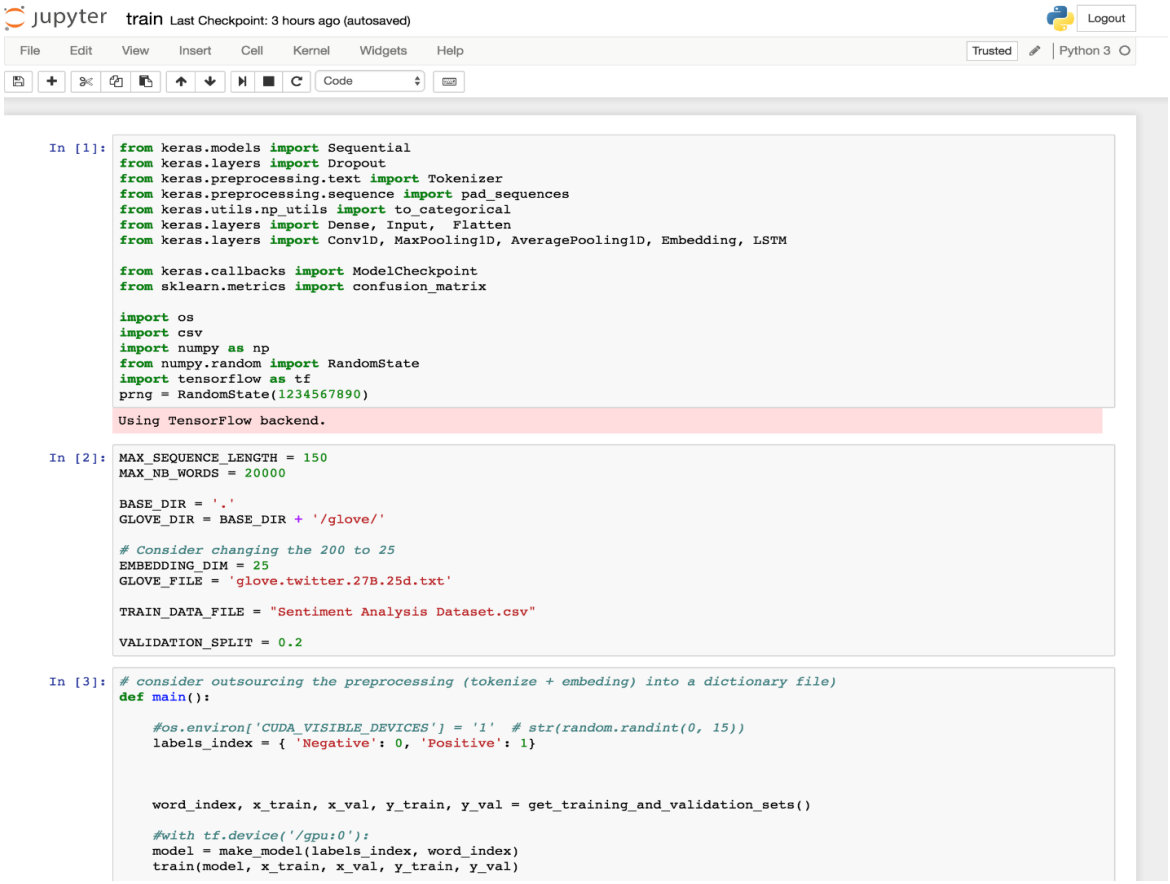

Figure 6. Screenshot for sample training code in jupyter notebook.

Following is the sample training code (Fig. 6) and running result (Fig. 7) with 5 epochs. (for all code please refer to the source code).

As we can see from Fig. 7, after 5 epochs, the training accuracy is 78.5% while val_acc is 80. For testing data, 121808 are negative true and 35622 are negative false; 27307 are positive false and 130986 are positive true. So, the total accuracy is (121808 + 130986) / (121808 + 35622 + 27307 + 130986) = 80%, which is relatively reliable for sentiment analysis.

For real tweets testing, we use two examples, and the results (Fig. 8) are reasonable.

```
Train
Train on 1262892 samples, validate on 315723 samples
Epoch 1/5
1262892/1262892 [==============================] - 1243s - loss: 0.5421 - acc: 0.7208 - val_loss: 0.4955 - val_acc:
0.7554
Epoch 2/5
1262892/1262892 [==============================] - 1196s - loss: 0.4903 - acc: 0.7601 - val_loss: 0.4558 - val_acc:
0.7825
Epoch 3/5
1262892/1262892 [==============================] - 1236s - loss: 0.4712 - acc: 0.7727 - val_loss: 0.4404 - val_acc:
0.7928
Epoch 4/5
1262892/1262892 [==============================] - 1225s - loss: 0.4603 - acc: 0.7799 - val_loss: 0.4300 - val_acc:
0.7994
Epoch 5/5
1262892/1262892 [==============================] - 1226s - loss: 0.4526 - acc: 0.7850 - val_loss: 0.4266 - val_acc:
0.8007
[[121808  35622]
 [ 27307 130986]]
```

Figure 7. Screenshot for sample training results.

```
[ Postive ]  A new platform developed by the World Economic Forum is unlike any visualization you've seen [ 0.3269768
4  0.6730231 ]
[ Neutral ]  Every woman who comes forward deserves to be heard, fully and completely, and our relationship to the ac
cused should not be part of the calculation anyone makes when examining her case [ 0.64857233  0.35142764]
```

Figure 8. Screenshot for real tweet test.

The following figured are results and analysis for the program we developed. Running the program is standard, it could be executed from command lines, or from various python IDE. After setting the search keywords, we are ready to run our program. Tweets start to flood into the terminal screen. We processed the "text" attribute so that we are only doing sentiment analysis on original tweets, which means no retweets are going to show oh the terminal screen (figure. 10). We did a test run for search the keyword "vacation", and we let our program run for 20 minutes, results are shown in the figure in real time (figure. 11). In the most recent 225 tweets that included keyword "vacation", 39.6% of the users were sharing positive feelings, while only 9.3% of the users were feeling negative about "vacation". This result makes perfect sense. After all, who wouldn't love vacations?

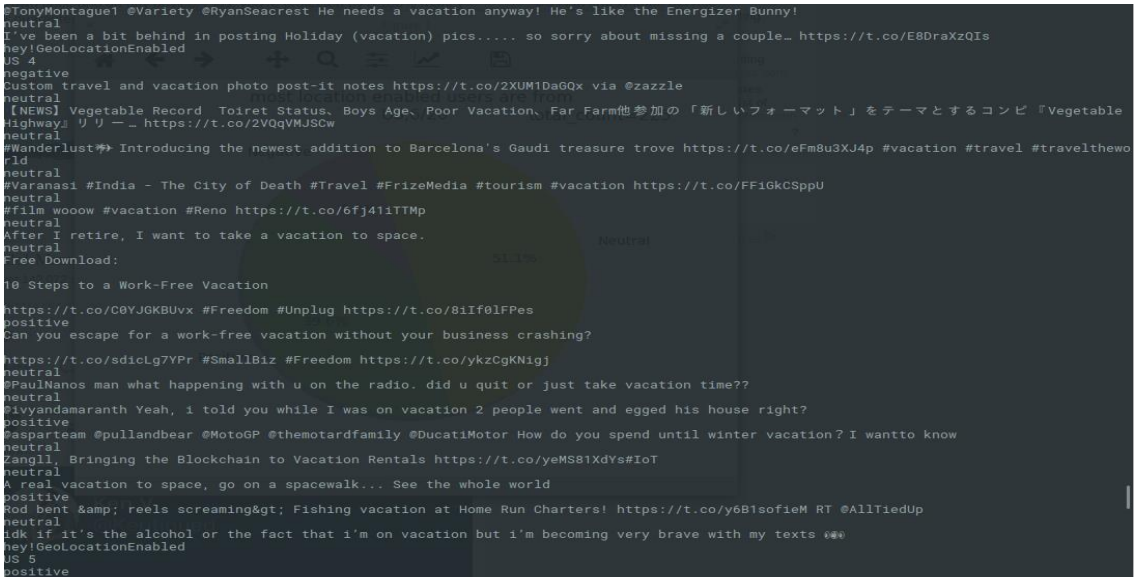

Figure 10. Screenshot for twitter keyword search "vacation".

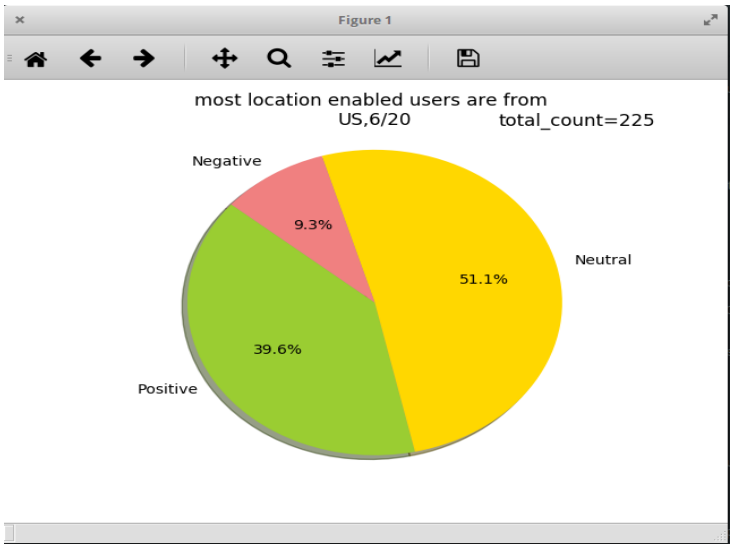

Figure 11. Screenshot for how twitter user's reaction on "vacation", and where they are from.

We now move on to more controversial topic. We chose the keyword to be bitcoin, its value just reached a new high of $7819. Base on many posted blogs, articles, analysis that we have read, it is not clear how people feel about bitcoin.

While this is a trending topic, we would like to know how people on twitter feels about bitcoin. After running our program for 15 minutes, we gathered the most recent 1000 tweets.

```
https://t.co/4GoLUpviUA
positive
Coinbase Custody Targets $10 Billion in Institutional Funds for Bitcoin Investment - https://t.co/ohl5nldhYL https://t.co/wm9vUS9lF6
neutral
Capital:
Bitcoin:$130,258,040,662
Ethereum:$32,879,062,815
Bitcoin Cash:$20,856,022,421
Ripple:$8,826,566,859
Litec… https://t.co/yPXICvl9ZO
neutral
Bitcoin: Could It Be Damaging to the Environment? https://t.co/ZJYfFJGSFA
neutral
SMS Coin is planning on being listed on exchanges in UK, Asia, and Latin America on Decem.. #bitcoin #blockchain https://t.co/xqrlwJ6qTx
neutral
FreeBitco.in - Free Bitcoin Wallet, Faucet, Lottery and Dice! https://t.co/UoQiBbL5ck
positive
$232m blockchain startup Tezos faces sueballs for alleged investor fraud • The Register #Blockchain #Bitcoin #btc #Ethereum #ETH -
negative
Six Exchange AG Offers New Speculative Certificate for Bitcoin Price Watchers https://t.co/tAZQtQLX7H
positive
An International Blockchain &amp;amp; Bitcoin Conference will take place in Slovenia! https://t.co/cNBNv6jcmW
neutral
@Daniel Reichmann @M SHAKER, @Bitcoin ICO Bounty @Mario Bazán
#ThanksFriend
#SaturdayFever #GoinDown
#Waterboy… https://t.co/yVSepSqSHw
neutral
Trade Recommendation: Bitcoin | https://t.co/JEd5wMVpJp - https://t.co/Hua4QecI79

Bitcoin's swift recovery was th… https://t.co/hyQ5uRkb35
neutral
@MrStephenHowson Stormzy_blaz
For all those interested in Bitcoin go to this site.

https://t.co/4GoLUpviUA
positive
Yeni vidyo, Bitcoin la nasil para yapabilirim?? Gel anlatiyorum..https://t.co/MzqIibqzxz
neutral
Rimbit on #Cryptopia - #RBT the only coin not mined- #RBT for #LTC https://t.co/vG0Kudm4Lh - #crypto #bitcoin… https://t.co/NBJ9PTkHhQ
neutral
Bitcoin is hitting new highs–here's why it might not be a bubble https://t.co/y0G6AWpJ1u
positive
Is this Bitcoin?: https://t.co/6fgFK08jvs via @YouTube
neutral
```

Figure 12. Screenshot for twitter keyword search "bitcoin".

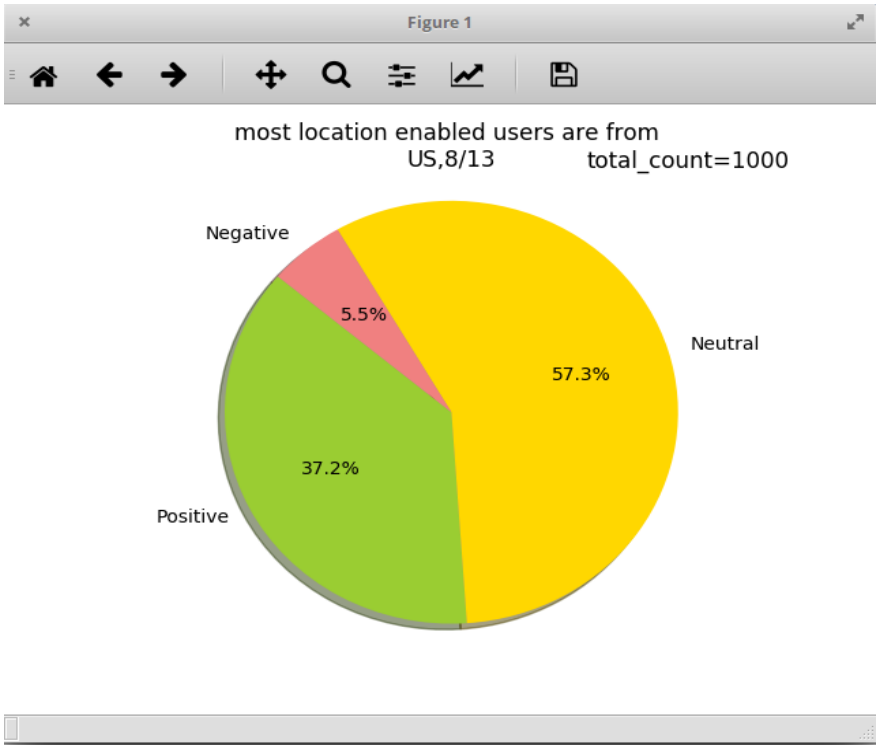

Figure 13. Screenshot for how twitter user's reaction on "bitcoin", and where they are from.

While we were monitoring the real time figure (figure 13) about how people feel about bitcoin, we quickly noticed that there are only about 5% of people thinking negatively on twitter. Apparently at this stage, twitter users like the result they were seeing in bitcoin, and we can predict that bitcoin's value are most likely continue to rise in the near future.

5. CONCLUSION

This real time twitter sentiment analysis project has practical use. In fact, there are some well developed products out there, which has the same idea and purpose, for user to subscribe and use, tweetping.net is a real word example. In order to add more analysis features for users, we could simply extract more features from twitter JSON and further process it. For example, if we are interested in at what duration of time in a day, that users sends the most tweets about certain keywords, we can extract the tweet creation time by looking for the attribute "created_at" in the twitter JSON, and the value will be the UTC time when the twitter was sent, and we then process this information. We could do the same to analyze tweet text language used, how many time a tweet has been retweeted, even apply image classification to the tweeted images. There is huge potential in this project. It not only enabled the capability for users to feel how trending topic impact people's opinion, but also let users to foresee situations and act upon early.

With a tagline, "What's happening in the world and what people are thinking about right now" , Twitter is the enclave of opinions, thoughts and beliefs, in other words, "sentiments". Every word, video, photo and follower can have an impact and this project, real time sentiment analysis, shows its potential for usage for practical purposes. Opinion analysis has its use in many domains, for identifying and cataloging a piece of text according to the tone conveyed by it. Sentiment analysis in any system can prove a major breakthrough for the complete brand revitalization. It has found its necessity in medical settings as well. Medical opinions concerning the patient's health status, medical conditions and treatment could characterize the facets of sentiment in the medical sphere and identify potential use cases.

All this also arises a question of accuracy which as we know can never be practically 100%. A constructed model can never comprehend sarcasm but then even people do not agree 80% of the time. This means even if a machine does not score a 100% it will still have more accuracy and hence credibility than human analysis. Also, when the corpus is huge, manual assessment is not an option. Which again makes sentiment analysis not just a trend but an asset. The applications of sentiment analysis can never be overlooked, mostly because of its ability of harnessing the plethora of unstructured data for actionable insights. The applications are definitely plenty and overwhelming, and as every budding application or technology it depends on what tool we use and how well we use it to our advantage.

REFERENCES

[1] Cs.cornell.edu. (2017). *Cite a Website - Cite This For Me*. [online] Available at: http://www.cs.cornell.edu/home/llee/omsa/omsa-published.pdf [Accessed 11 Nov. 2017].

[2] Prager, J. (2017). *Open-Domain Question–Answering*.

[3] R. Socher, Y. Bengio and C. Manning, "Deep learning for NLP (without magic)", Proceeding ACL '12 Tutorial Abstracts of ACL 2012, 2012.

[4] B. Krishnamurthy, N. Puri and R. Goel, "Learning Vector-space Representations of Items for Recommendations Using Word Embedding Models", Procedia Computer Science, vol. 80, pp. 2205-2210, 2016.

[5] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke and J. Schmidhuber, "A Novel Connectionist System for Unconstrained Handwriting Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 5, pp. 855-868, 2009.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[7] "Understanding LSTM and its diagrams – ML Review – Medium", Medium, 2017. [Online]. Available: https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714. [Accessed: 20- Nov- 2017].

[8] Scikit-learn.org. (2017). [online] Available at: http://scikit-learn.org/stable/modules/naive_bayes.html [Accessed 9 Nov. 2017].

[9] Alumni.media.mit.edu. (2017). [online] Available at: http://alumni.media.mit.edu/~hugo/publications/papers/CICLING2006-nlp4nlp.pdf [Accessed 18 Nov. 2017].

[10] Textblob.readthedocs.io. (2017). *TextBlob: Simplified Text Processing — TextBlob 0.13.1 documentation*. [online] Available at: https://textblob.readthedocs.io/en/dev/ [Accessed 19 Nov. 2017].

[11] "Amazon EC2 Instances - Amazon Elastic Compute Cloud", Docs.aws.amazon.com, 2017. [Online]. Available: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Instances.html. [Accessed: 20- Nov- 2017].

[12] "AWS Marketplace: Bitfusion Ubuntu 14 TensorFlow", Aws.amazon.com, 2017. [Online]. Available: https://aws.amazon.com/marketplace/pp/B01EYKBEQ0. [Accessed: 20- Nov- 2017].

[13]"Keras Documentation", Keras.io, 2017. [Online]. Available: https://keras.io. [Accessed: 20- Nov- 2017].

[14] "Twitter Sentiment Analysis Training Corpus (Dataset)", Thinknook, 2017. [Online]. Available: http://thinknook.com/twitter-sentiment-analysis-training-corpus-dataset-2012-09-22/. [Accessed: 20- Nov- 2017].

[15] J. Pennington, "GloVe: Global Vectors for Word Representation", Nlp.stanford.edu, 2017. [Online]. Available: https://nlp.stanford.edu/projects/glove/. [Accessed: 20- Nov- 2017].

[16]"Project Jupyter", Jupyter.org, 2017. [Online]. Available: http://jupyter.org. [Accessed: 20- Nov- 2017].

[17] SMITH, K. (2016). 44 Twitter Statistics for 2016. [online] brandwatch. Available at: https://www.brandwatch.com/blog/44-twitter-stats-2016/ [Accessed 16 Nov. 2017].

[18] Twitter Developer. (2017). Introduction to Tweet JSON. [online] Available at: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json [Accessed 16 Nov. 2017].

[19] Tweepy, (2017). Tweepy Documentation. [online] Available at: http://tweepy.readthedocs.io/en/v3.5.0/getting_started.html#introduction [Accessed 14 Nov. 2017].

[20] Matplotlib, (2017). matplotlib documentation. [online] Available at: https://matplotlib.org/ [Accessed 20 Nov. 2017].

[21] Unamo.com. (2017). *Understanding Sentiment Analysis in Social Media Monitoring | Unamo Blog*. [online] Available at: https://unamo.com/blog/social/sentiment-analysis-social-media-monitoring [Accessed 10 Nov. 2017].

[22] Ieeexplore.ieee.org. (2017). *Real time sentiment analysis of tweets using Naive Bayes - IEEE Conference Publication*. [online] Available at: http://ieeexplore.ieee.org/document/7877424/?reload=true [Accessed 4 Nov. 2017].