TWITTER EMOTION ANALYSIS

By

HARI KRISHNA N 2015103017

A project report submitted to the

FACULTY OF INFORMATION AND

COMMUNICATION ENGINEERING

in partial fulfillment of the requirements for
the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND

ENGINEERING

ANNA UNIVERSITY, CHENNAI – 25

MAY 2013

ABSTRACT

These days, Social networking sites like twitter, Facebook, etc. are the great source of communication for internet users. So these become an important source for understanding the opinions, views or emotions of people. In this paper, we use data mining techniques for the purpose of classification to perform emotion analysis on the views people have shared on Twitter, which is one of the most used social networking sites nowadays.

We collect dataset, i.e. tweets from Twitter and apply text mining techniques – transformation, tokenization, stemming etc. to convert them into a useful form and then use it for building emotion classifier. Here, we are using different classifiers on the data and then compare the results to find which one gives better accuracy and better results.

Table of Contents

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Problem domain:

In the current world of business analytics, analysts are constantly trying to identify the information about their users that can help them with providing better services. But to collect this information, which has to be credible and reliable, it's not a very easy task. Today with microblogging websites like Twitter which provides developers to collect the information of the users, we can easily collect the information of the users and perform emotion analysis and figure what the general audience feel about any particular problem they face or any product in the market.

1.2 Problem Description:

Given any tweet crawled from the internet this model should be able to analyze the given tweet and determine what emotion is associated with the tweet or what the user wants to convey through his tweet, be it a happy emotion or a sad emotion. By training a model with a given dataset and using the model to test further data to determine its accuracy is the goal of this project.

1.3 Scope:

There are various micro blogging sites in this fast paced world where information is constantly exchanged over these sites and a lot of data about the people and their needs are available to all the multinational corporates. By analyzing this data that is freely available to us we can easily help the people of this world and make their lives easier in whatever way suits them.

1.4 Contribution:

With this project, I am adding a different look at text mining by using RNN models which are usually preferred for image classification. By learning and understanding the user data at a fundamental level I am trying to one up the already existing models and produce a model with higher accuracy.

1.5 SWOT Analysis:

1.5.1 Strength

There are multiple strengths to the idea of mining data from microblogging sites. We get real time word about the user's opinions and their personal views on many issues going around the world. By using this data, we can predict the main problems of the masses of the world and try to repair it. Also we can read into problems well before they become a huge problem and can quickly act upon it. The method implemented in this thesis is very high in prediction rate also which it makes it more accurate.

1.5.2 Weakness

The main weakness of this project is the accuracy of the prediction. In the case where the accuracy rate is poor, then it becomes hard to go forward with tackling these issues as we don't know the exact magnitude of the problem. For example, if our system reports 5 people are unhappy with something and classifies the issue as unhappy then it becomes a problem. Because 10 other people may be happy with the issue. Hence we must be careful.

1.5.3 Opportunities

As mentioned in the problem domain, the main attribute of this project is that we can use the already available data to predict the future happenings. The MNC's across the world can take advantage of this and use it to improve their profit margins. We can improve this model by using better algorithms to predict the data.

1.5.4 Threats

The only major threat associated with this project is the issue of privacy. People's everyday statements are continuously scrutinized and are taken advantage of without their authorization. So if anyone with malicious intent can misuse the data in a wrong manner.

1.6 PESTEL Analysis

1.6.1 Political

With the abundant data available, government organizations can use it to address the problems that happen at a national level. For example, demonetization came with a lot of problems. People took to posting their issues on the micro-blogging websites and the government can use it to analyze this to reach out to the people and sole their issues.

1.6.2 Economic

The most advantage from this type of data mining can be achieved in the economic thick of things. By predicting the user's needs and requirements through their online comments we can easily see into the future and go ahead with producing goods in large. The companies which can see this opportunity first receives maximum advantage.

1.6.3 Social

Also known as socio-cultural factors, are the areas that involve the shared belief and attitudes of the population. These factors include – population growth, age distribution, health consciousness, and career attitudes and so on. These factors are of particular interest as they have a direct effect on how marketers understand customers and what drives them.

1.6.4 Technological

The Anomaly detection and categorization mechanism employs long short term memory (LSTM), a state of the art Deep Learning technique for performing Twitter Emotion Analysis. Both techniques are the most widely used and dependable algorithmic solutions available to suit the need for such intense and voluminous data processing and feature selection. SentiWordNet is the dataset that is used for training and testing the model. The application can be migrated to a better and more efficient model in the future which showcases the level of flexibility the problem possesses. The use of Deep Learning allows the model to scale as per the amount of data given for processing.

1.6.5 Environmental

The applied ideologies and strategies bear no harm to the deployed environment. Power consumption would be a drawback considering the complexity of the model and the level of hardware it would require for smooth functioning.

1.6.6 Legal

Legal factors include - health and safety, equal opportunities, advertising standards, consumer rights and laws, product labelling and product safety. It is clear that companies need to know what is and what is not legal in order to trade successfully. If an organization trades globally this becomes a very tricky area to get right as each country has its own set of rules and regulations.

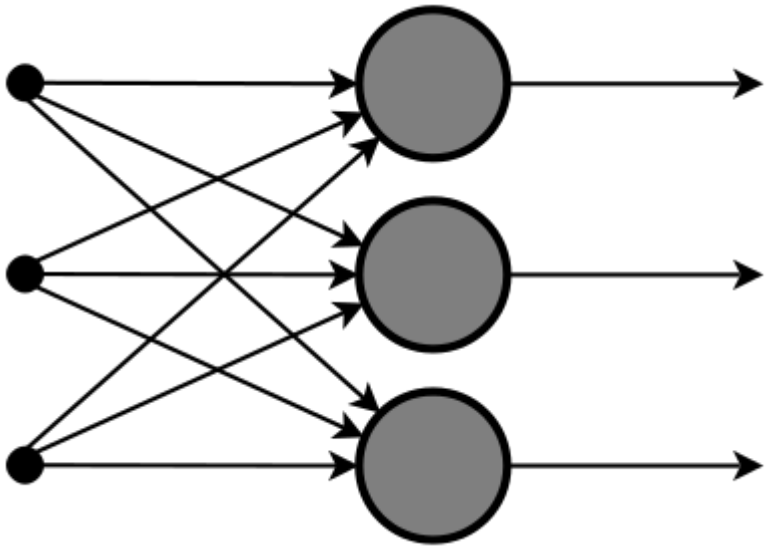
CHAPTER 2

RELATED WORKS

This chapter gives insight into the different methodologies for performing Emotion analysis from microblogging sites like Twitter, Reddit, etc. 4 of these methods were based on the Research paper that we referenced in a quest to find the proper technique that we could use for the problem at hand. This survey helped us evaluate each method on several parameters.

2.1 Feed forward Neural Network – Artificial Neuron:

This neural network is one of the simplest form of ANN, where the data or the input travels in one direction. The data passes through the input nodes and exit on the output nodes. This neural network may or may not have the hidden layers. In simple words, it has a front propagated wave and no back propagation by using a classifying activation function usually. Below is a Single layer feed forward network. Here, the sum of the products of inputs and weights are calculated and fed to the output. The output is considered if it is above a certain value i.e. threshold (usually 0) and the neuron fires with an activated output (usually 1) and if it does not fire, the deactivated value is emitted (usually -1). Application of Feed forward neural networks are found in computer vision and speech recognition where classifying the target classes are complicated. These kind of Neural Networks are responsive to noisy data and easy to maintain. This paper explains the usage of Feed Forward Neural Network. The X-Ray image fusion is a process of overlaying two or more images based on the edges. Here is a visual description.



2.1 Artificial Neural Network

2.2 Recurrent Neural Network (RNN)

The Recurrent Neural Network works on the principle of saving the output of a layer and feeding this back to the input to help in predicting the outcome of the layer. Here, the first layer is formed similar to the feed forward neural network with the product of the sum of the weights and the features. The recurrent neural network process starts once this is computed, this means that from one time step to the next each neuron will remember some information it had in the previous time-step. This makes each neuron act like a memory cell in performing computations. In this process, we need to let the neural network to work on the front propagation and remember what information it needs for later use. Here, if the prediction is wrong we use the learning rate or error correction to make small changes so that it will gradually work towards making the right prediction during the back propagation.

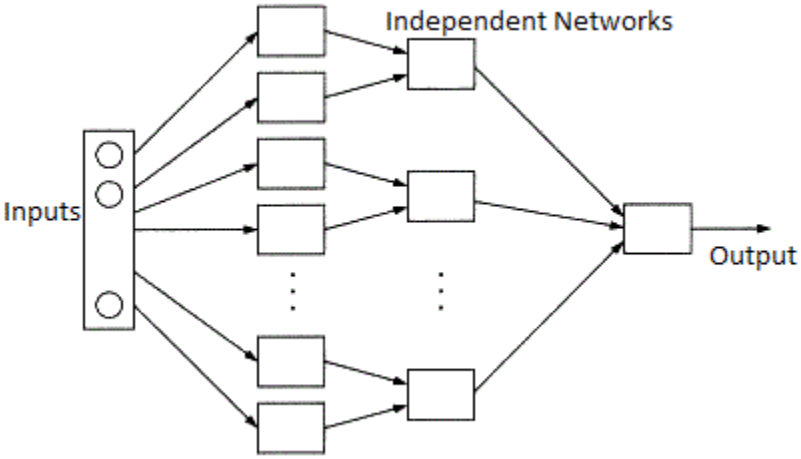
2.3 Convolutional Neural Network

Convolutional neural networks are similar to feed forward neural networks, where the neurons have learn-able weights and biases. Its application have been in signal and image processing which takes over OpenCV in field of computer vision. Below is a representation of a ConvNet, in this neural network, the input features are taken in batch wise like a filter. This will help the network to remember the images in parts and can compute the operations. These computations involve conversion of the image from RGB or HSI scale to Gray-scale. Once we have this, the changes in the pixel value will help detecting the edges and images can be classified into different categories. ConvNet are applied in techniques like signal processing and image classification techniques. Computer vision techniques are dominated by convolutional neural networks because of their accuracy in image classification. The technique of image analysis and recognition, where the agriculture and weather features are extracted from the open source satellites like LSAT to predict the future growth and yield of a particular land are being implemented.

2.4 Modular Neural Network

Modular Neural Networks have a collection of different networks working independently and contributing towards the output. Each neural network has a set of inputs which are unique compared to other networks constructing and performing sub-tasks. These networks do not interact or signal each other in accomplishing the tasks. The advantage of a modular neural network is that it breakdowns a large computational process into smaller components decreasing the complexity. This breakdown will help in decreasing the number of connections and negates the interaction of these network with each other, which in turn will increase the

computation speed. However, the processing time will depend on the number of neurons and their involvement in computing the results. Modular Neural Networks (MNNs) is a rapidly growing field in artificial Neural Networks research. This paper surveys the different motivations for creating MNNs: biological, psychological, hardware, and computational. Then, the general stages of MNN design are outlined and surveyed as well, viz., task decomposition techniques, learning schemes and multi-module decision-making strategies**.**



2.2 Modular Neural Network

2.5 Major NLP Areas Researched

This work is connected to five different areas of NLP research, each with their own large amount of related work to which we cannot do full justice given space constraints.

Semantic Vector Spaces. The dominant approach in semantic vector spaces uses distributional similarities of single words. Often, co-occurrence statistics of a word and its context are used to describe each word (Turney and Pantel, 2010; Baroni and Lenci, 2010), such as tf-idf. Variants of this idea use more complex frequencies such

as how often a word appears in a certain syntactic context (Pado and Lapata, 2007; Erk and Pado, 2008). However, distributional vectors often do not properly capture the differences in antonyms since those often have similar contexts. One possibility to remedy this is to use neural word vectors (Bengio et al., 2003). These vectors can be trained in an unsupervised fashion to capture distributional similarities (Collobert and Weston, 2008; Huang et al., 2012) but then also be fine-tuned and trained to specific tasks such as sentiment detection (Socher et al., 2011b). The models in this paper can use purely supervised word representations learned entirely on the new corpus.

Compositionality in Vector Spaces. Most of the compositionality algorithms and related datasets capture two word compositions. Mitchell and Lapata (2010) use e.g. two-word phrases and analyze similarities computed by vector addition, multiplication and others. Some related models such as holographic reduced representations (Plate, 1995), quantum logic (Widdows, 2008), discrete-continuous models (Clark and Pulman, 2007) and the recent compositional matrix space model (Rudolph and Giesbrecht, 2010) have not been experimentally validated on larger corpora. Yessenalina and Cardie (2011) compute matrix representations for longer phrases and define composition as matrix multiplication, and also evaluate on sentiment. Grefenstette and Sadrzadeh (2011) analyze subject-verb-object triplets and find a matrix-based categorical model to correlate well with human judgments. We compare to the recent line of work on supervised compositional models. In particular we will describe and experimentally compare our new RNTN model to recursive neural networks (RNN) (Socheret al., 2011b) and matrix-vector RNNs (Socher et al., 2012) both of which have been applied to bag of words sentiment corpora.

Logical Form. A related field that tackles compositionality from a very different angle is that of trying to map sentences to logical form (Zettlemoyer and Collins, 2005). While these models are highly interesting and work well in closed domains and on discrete sets, they could only capture sentiment distributions using separate mechanisms beyond the currently used logical forms.

Deep Learning. Apart from the above mentioned work on RNNs, several compositionality ideas related to neural networks have been discussed by Bottou (2011) and Hinton (1990) and first models such as Recursive Auto-associative emories been experimented with by Pollack (1990). The idea to relate inputs through three way interactions, parameterized by a tensor have been proposed for relation classification (Sutskever et al., 2009; Jenatton et al., 2012), extending Restricted Boltzmann machines (Ranzato and Hinton, 2010) and as a special layer for speech recognition (Yu et al., 2012).

Sentiment Analysis. Apart from the above mentioned work, most approaches in sentiment analysis use bag of words representations (Pang and Lee, 2008). Snyder and Barzilay (2007) analyzed larger reviews in more detail by analyzing the sentiment of multiple aspects of restaurants, such as food or atmosphere. Several works have explored sentiment compositionality through careful engineering of features or polarity shifting rules on syntactic structures (Polanyi and Zaenen, 2006; Moilanen and Pulman, 2007; Rentoumi et al., 2010; Nakagawa et al., 2010).

2.6 Observations from the Survey

From the various methods that we surveyed it was clear that most of the methods employed naïve machine learning approaches. Only Long Short Term Model proved to be a standout method given the fact that it was far more advanced in terms of its architecture compared to the other six machine learning approaches.

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 Functional Requirements

The system should be able to correctly predict what emotion is associated with any input tweet fed into it. It should do the following things:

- Stream the twitter data live using the available API
- Pre-process the tweet removing all unnecessary characters
- Build a training model based off the tweets collected
- Correctly predict any incoming tweet and associate an emotion to it.
- Use emotion to identify to understand users emotion towards anything.

3.2 Nonfunctional Requirements

3.2.1 User Interface:

The user requires a readable interface where the input can be fed into the application and the generated output can be viewed, interpreted visually and saved for later.

3.2.2 Hardware:

No particular additional hardware is required for this project. A simple 64-bit processor laptop or desktop is enough to complete this project.

3.2.3 Software:

Operating system: Windows

Programming Language: Python

Twitter API

Tools: Anaconda Navigator

3.2.4 Performance:

The system's performance is stable, optimized and consistent but requires Graphical Processing Units.

3.3 Constraints

- The system only considers text and removes any other form text from the tweets. E.g. Emoticons.
- Not all tweets are clear and some of them lead to ambiguous or neutral statements.
- No particular UI is done to show the output. It is saved as a CSV file.

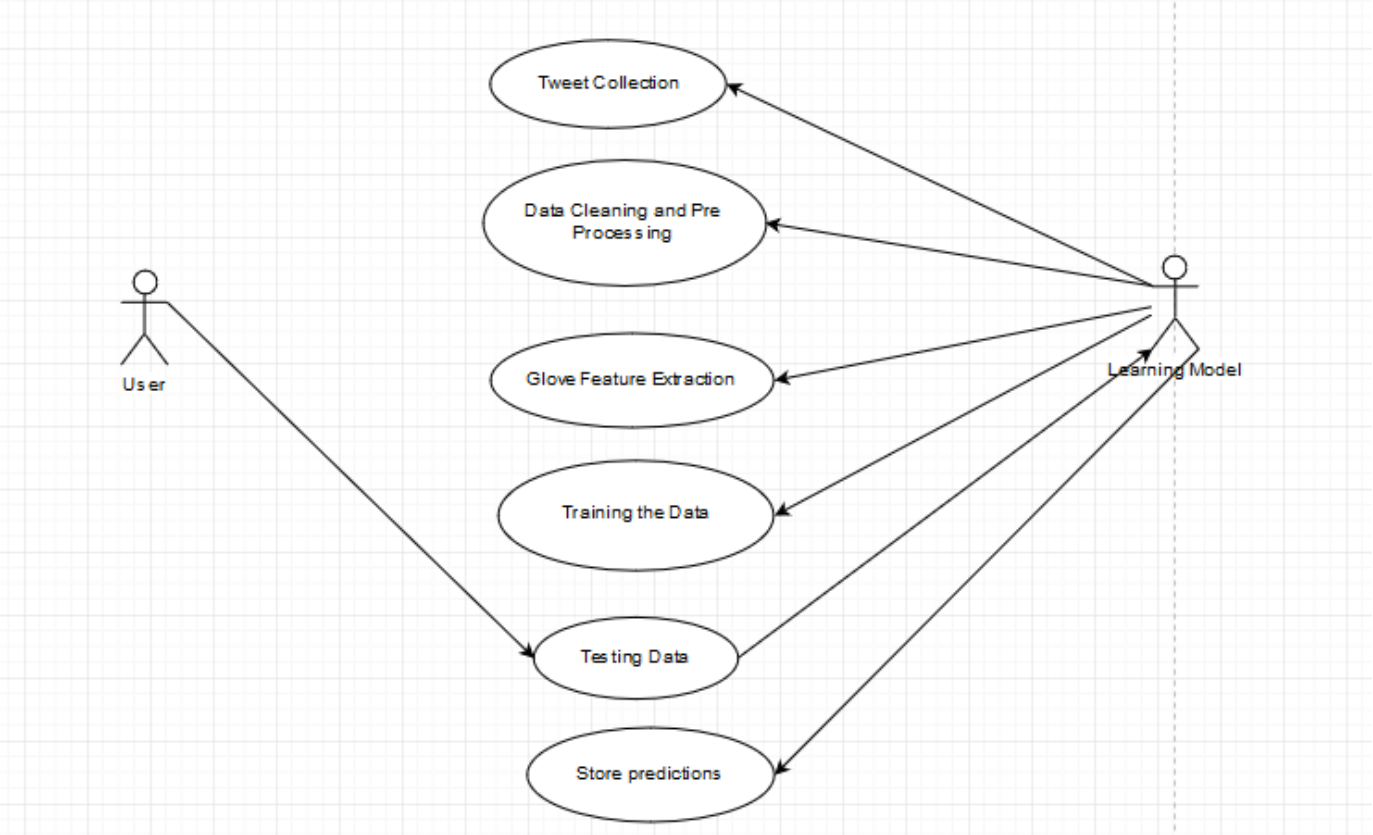
3.4 Assumptions

- The number of tweets used to determine this is minimal. For higher accuracies more tweets can be considered.
- Not all tweets have an emotion. Some might just be general facts.
- We are only assuming two types of emotion for this project: Happy and Sad.

3.5 System Models

3.5.1 Over-all Use Case Diagram

The overall use case diagram for the Twitter emotion Analysis is given below.



3.1 Use case Diagram

Pre-condition: An input tweet is given by the user.

Post-condition: The predicted result is stored in a CSV file.

CHAPTER 4

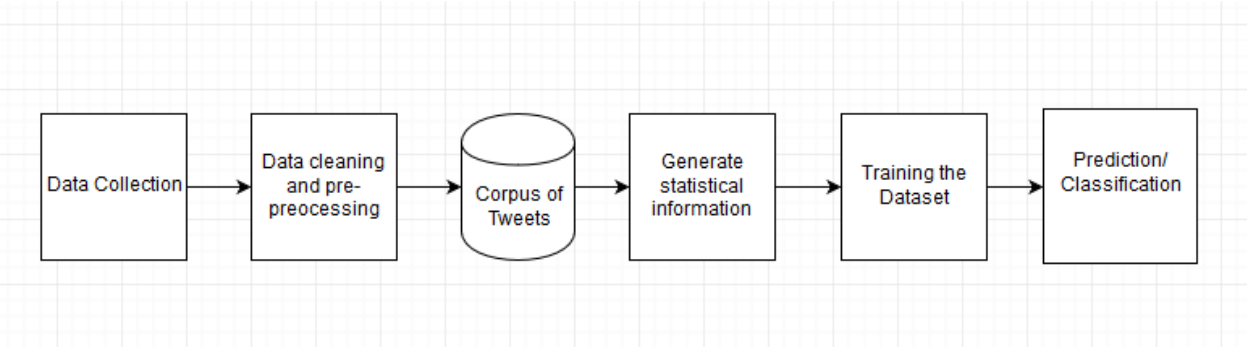
SYSTEM DESIGN

4.1 System architecture

The block diagram has been shown below. The Tweepy API has been used to crawl the data from the Twitter account using the credentials of a developer account. This API was chosen over the popular API because this is closer to the Twitter API and provides easier iteration.

It uses regular expressions to clean and process the incoming data that are stored in a CSV file. CSV files are used because it is easier to iterate through different columns separated by commas. All the extra materials other than the text such as the URLs, Emoticons extra are removed. We also use Porter Stemmer to remove the stem words from the incoming tweets.

After pre-processing the data we also calculate the basic statistical information about the data such as the unigrams and bi-grams and store them in pickle files as they are easier to work with.



4.1 System architecture

After all the pre-processing work is done, we come to training the dataset. In order to do this we generate the glove. GloVe, coined from Global Vectors, is a model for distributed word representation. The model is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Then we start training the dataset by setting all the initial conditions. Since we are using the RNN classifier we set all the default conditions required for this classifier. We split the data in 90-10 split ratio and train 90% of the data and use the remaining 10 % to determine the accuracy. After training, we can feed in our own data and get predictions for what type of emotions they are.

4.2 UI Design

A simple and easy to use User Interface has been designed for the system using the terminal. The user inputs their query in a csv file. And the answer is displayed in another csv file.

4.3 Module Design

4.3.1 Data Collection

The data collection is done using the Tweepy API. In order to execute this API, we need a Twitter Development account. After creating the account, we get a few credentials which we use in our code and use it to crawl the data. After crawling,

the Tweet ID, the sentiment associated (1 if happy and 0 if sad), and the tweet itself is stored in CSV file.

4.3.2 Data Cleaning and Pre-Processing

After all the data has been crawled from the Twitter account and stored in the CSV file, we need to clean it. To clean a Tweet means to remove all the elements in the Tweet which won't contribute to the context of the tweet itself. URLs, emoticons, special characters, etc. are examples of this. So we use regular expressions to parse the Tweets from the CSV file and remove all these unnecessary elements. After removing it, we also use Porter Stemmer in order to remove the stop words from the tweets. So all the insignificant words in English which are just used as sentence connectifiers are removed.

4.3.3 Generate Statistical Information

We generate all the basic statistics about our data. This includes the user mentions, the number of URLs, Emojis, etc. Along with the other basic information about our dataset, we also calculate the unigrams and the bigrams of our dataset. Unigrams are the number of individual words in our dataset. Bigrams are the contiguous sequence of 2 items from a given sample of text or speech. By calculating the unigrams and the bigrams we can use it to calculate the probability of two words occurring together or the type of words used in our corpus of data.

4.3.4 Training the Model

This is the main part of our system. We use the LSTM model or the more commonly known Recurrent Neural Network model to train the data set. A recurrent neural network (RNN) is a class of artificial neural network where connections between nodes form a directed graph along a sequence. This allows it to exhibit

temporal dynamic behavior for a time sequence. Unlike feed forward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition] or speech recognition.

So in our case, we use the generated statistical file from the training data set and our dataset itself to train the model using by initializing multiple parameters and using the Keras package. Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit or Theano. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. The model trains the dataset through 5 iterations splitting the dataset in 90-10 split ratio and using the 90% data to learn the dataset and use the remaining 10 to predict the accuracy.

4.3.5 Prediction/Classification

For prediction we need to change the model from a TRAIN type to a TEST type by setting a few parameters. Then we generate the general statistical information about the test data. By feeding both the statistics as a pickle file and the test data as CSV file to the system we can start using the trained model to make predictions. The predictions are stored in a separate CSV file containing 1s and 0s. Here 1s mean that the particular tweet is happy and 0 meaning the tweet is sad.

4.4 Complexity Analysis

4.5.1 Time complexity

The output can be read from the RNN after a number of time steps that is asymptotically linear in the number of time steps used by the Turing machine and asymptotically linear in the length of the input.

All the other modules use O (n) time complexity as they jus depend on the input size.

4.5.2 Complexity of the Project

A lot of complexities were faced in the making of this project. The major complexities are:

- All the scrapped Tweets have to be labelled manually for better accuracy in the training data.
- The accuracy of the algorithm also affects the accuracy of the prediction.
- Extra codes for generating thee statistical information about the data has to be generated.
- A lot of the tweets in the input data might be neutral. They have to be considered as sad tweets because separate classification would take time.

CHAPTER 5

SYSTEM DEVELOPMENT

The system described consists of various packages like Keras, CSV, Utils, Numpy, Sci-kit Learn, etc. The overall code overview showing the organization of these various packages of the Machine Translation system can be seen in figure.

```
import numpy as np
import sys
from keras.models import Sequential, load_model
from keras.layers import Dense, Dropout, Activation
from keras.layers import Embedding
from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau
from keras.layers import LSTM
import utils
from keras.preprocessing.sequence import pad_sequences
```

An overview of the algorithm of entire system is shown below. The input Tweet T is given to the LSTM model which produces the Emotion associated with it.

T ← Tweet

Predict (T)

Model(T) ← Emotion

5.1 Prototype across the Modules

The input and output to each module of the system is described in this section.

- **Data Collection**: This module uses the Tweepy API thereby taking in the query from the user to collect a certain type of data. It stores the tweet ID and the tweet in a CSV file.
- **Data Cleaning and Pre-Processing:** This module reads the tweets from the CSV file, uses regular expressions and porter stemmer to clean the tweets and strip it of all unwanted elements.
- **Generate Statistical Information:** This module uses the cleaned tweets from another CSV file and calculates the unigrams and bigrams and stores the result in a pickle file.
- **Training the Model:** This module is the heart of the system and uses the previously generated clean tweets files and the General Statistics file to train the model and store five iterations of the model of the model, 5^{th} being the most accurate, in a .hd5 file.
- **Prediction and Classification:** This module uses a dedicated testing dataset for this model and uses it to predict the emotion associated with the tweet and store it in a CSV file.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Dataset for Testing

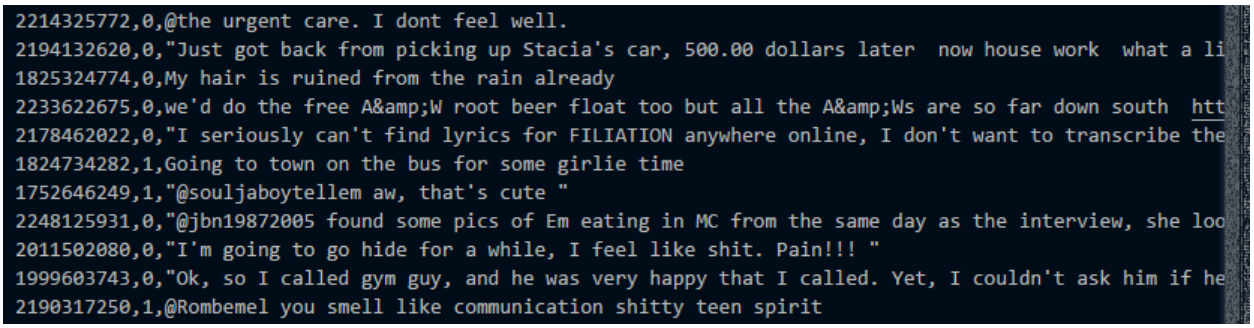
The dataset for testing is a CSV file containing two columns of value. The first column contains the tweet ID of the user who tweeted. The second column contains a cleaned version of the tweet using regular expressions and porter stemmer.

6.2 Output Obtained At Various Stages

This section shows the various intermediate results during module testing.

6.2.1 Initial Training Dataset

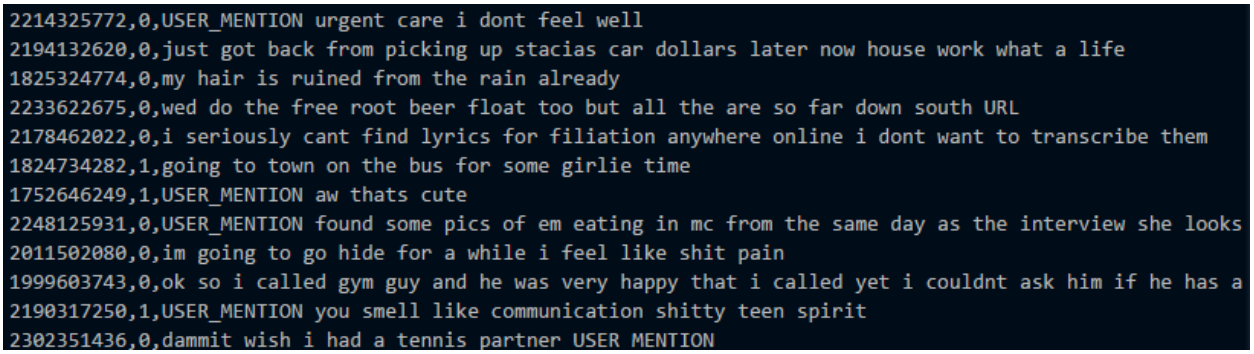
This is the output obtained after using the Tweepy API to collect the tweets.



6.1 Training Dataset

6.2.2 Cleaned Tweets

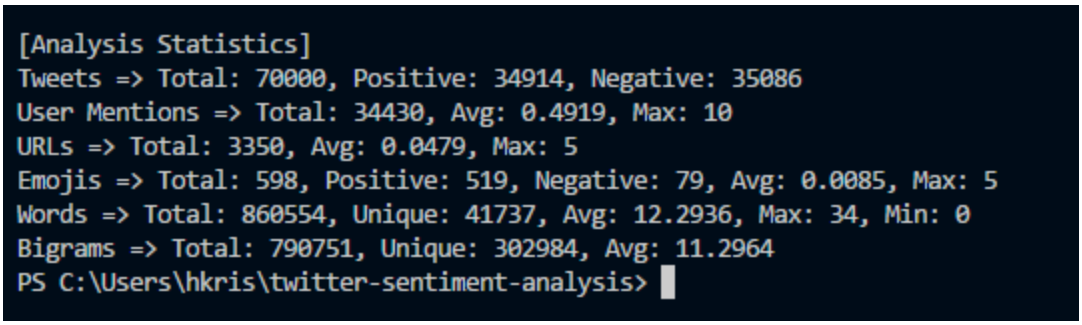
This is the CSV file after cleaning the tweets.



6.2 Cleaned Dataset

6.2.3 General Statistical Information

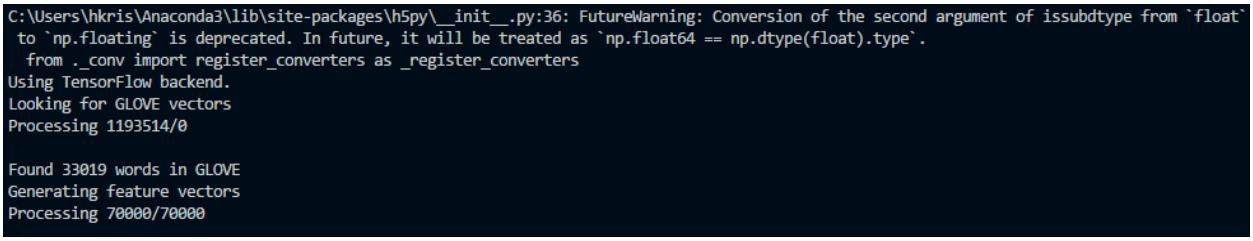
This is the general statistics generated for the dataset.



6.3 Statistics of the Training Dataset

6.2.4 Intermediate Results While Training the Model

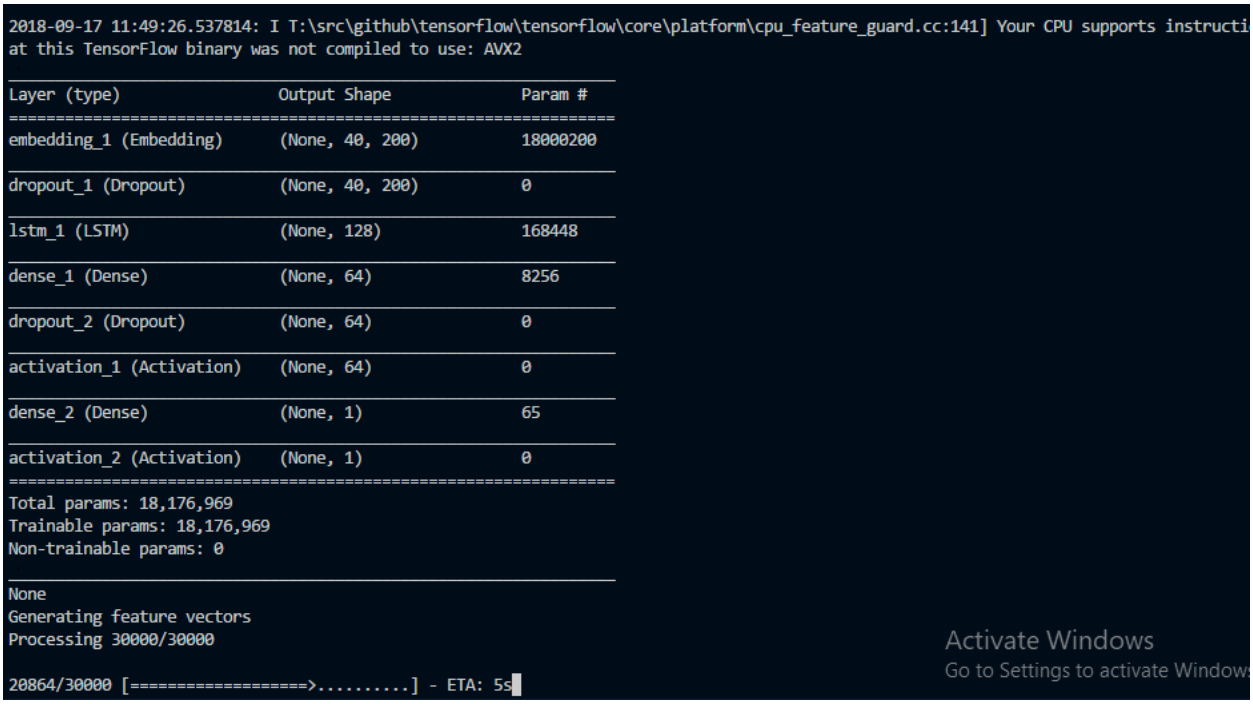
These are the intermediate results obtained while training the model.



6.4 Intermediate Screen while Training

6.2.5 Intermediate Result While Testing the Data

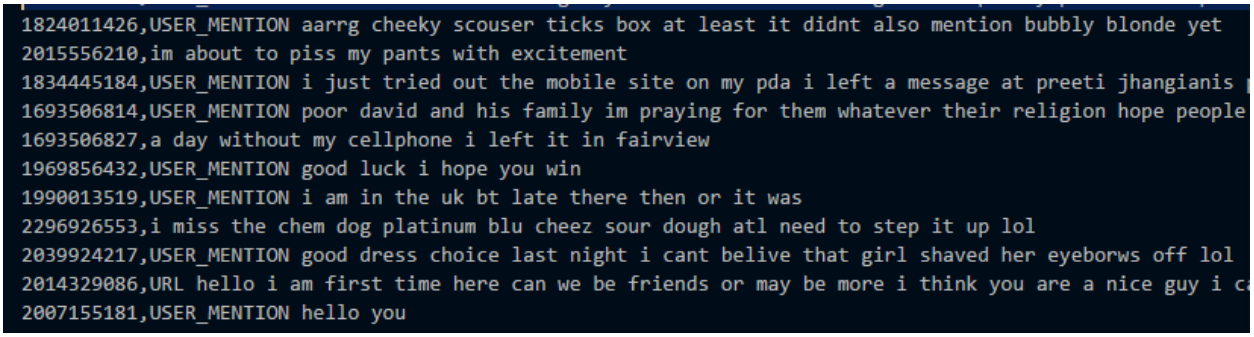
This shows the intermediate result obtained during testing the data.



6.5 Intermediate Screen while Testing

6.3 Sample Screenshots during Testing

A part of the input and he output are show in the below figures. The system is tested for 30000 different tweets.



6.6 Testing Dataset

6.7 Predictions for Tweets

6.4 Performance Evaluation

The performance of the system is evaluated using various parameters, including the precision, value loss, recall and the accuracy of the system. We also measure performance using parameters such as True Positive, True Negative, False positive and false negatives. In the case of predicting emotions we'd rather not have true negatives or false negatives as they will reduce the accuracy of the program.

6.4.1 True Negative Predictions

The possible example for true negative tweets are,

"I am not happy today #nothappy" being classified as 1, just because it have the word happy doesn't mean that the tweet is happy.

6.4.2 True Positive Predictions

The possible example for true positive tweets are,

"I am not happy today #nothappy" being classified as 0.

6.4.3 False Negative Predictions

The possible example for false negative tweets are,

"I am working today for 8 hours #work" being classified as 1, just because it have the word happy doesn't mean that the tweet is happy and is rather a neutral tweet.

6.4.4 False Positive Predictions

The possible example for false positive tweets are,

"I am not happy today #nothappy" being classified as 1, just because it has the word happy doesn't mean that the tweet is happy.

```
Train on 63000 samples, validate on 7000 samples
Epoch 1/5
63000/63000 [==============================] - 355s 6ms/step - loss: 0.5233 - acc: 0.7387 - val_loss: 0.4589 - val_acc: 0.7841

Epoch 00001: loss improved from inf to 0.52334, saving model to ./models/lstm-01-0.523-0.739-0.459-0.784.hdf5
Epoch 2/5
63000/63000 [==============================] - 339s 5ms/step - loss: 0.4450 - acc: 0.7965 - val_loss: 0.4499 - val_acc: 0.7893

Epoch 00002: loss improved from 0.52334 to 0.44501, saving model to ./models/lstm-02-0.445-0.797-0.450-0.789.hdf5
Epoch 3/5
63000/63000 [==============================] - 330s 5ms/step - loss: 0.4063 - acc: 0.8193 - val_loss: 0.4504 - val_acc: 0.8017

Epoch 00003: loss improved from 0.44501 to 0.40626, saving model to ./models/lstm-03-0.406-0.819-0.450-0.802.hdf5
Epoch 4/5
63000/63000 [==============================] - 323s 5ms/step - loss: 0.3700 - acc: 0.8392 - val_loss: 0.4514 - val_acc: 0.7930
```

6.8 Performance metrics

CHAPTER 7

CONCLUSIONS

7.1 Summary

This is a standard twitter emotion predictor which predicts whether the given tweet is either happy or sad. The training dataset is collected using the Tweepy API and labelled as either happy or sad (0 or 1). The dataset is then preprocessed and cleaned. The cleaning process includes stripping all the emoticons, URLs, and other stop words using regular expressions as well as porter stemmer in order to remove the stop words from the tweets. After cleaning the tweets, we calculate the basic statistical values of the tweets (number of tweets, number of words, unigram and bigram). All these values are stored in a pickle file so that they are preserved.

Next we train the model using the LSTM classifier. Before training there are a lot of prerequisites to be done. We need to create an embedding matrix where the GLOVE vectors are stored. GLOVE vectors are used for learning continuous-space vector representations of words. Then we extract the feature vectors from the tweet by splitting the tweet into words and appending it to a vector. Then using Keras package we create a model and use the training dataset to train the model. We use the sigmoid function as the activation function. We train the model using our preprocessed dataset and we store 5 different models. These models differ in accuracy by a small amount and we can choose the best model which gives highest accuracy to test our data.

For testing, we again preprocess the testing data and strip off all the unnecessary information in the tweet. Then we also calculate the basic statistical information such as the unigrams and bigrams which we will use when we test

our data. We load our best model from training, the one that has highest accuracy. We then process our tweets by padding them and then predicting the emotion associated with it using our trained model. The results are stored in CSV file as 1s and 0s (1 being Happy and 0 being Sad).

7.2 Criticisms

The dataset consists of neutral statements as well, which had to be manually removed. The major errors are caused due to the ambiguous nature of the tweets. Some might contain relatively happier words but are actually neutral messages. Other sad tweets may contain occasional happy words which cause the ambiguity. Also the dataset is relatively small. With a bigger dataset more accurate predictions can be made.

7.3 Future Works

The efficiency of the system is at a moderate level but with a better and more classified dataset, the results can be improved to a greater level of accuracy. Also using a different classifier can probably improve the accuracy of the whole system. We can also emoticons in our future to get the full essence of what the tweet really means (for example sarcasm). We can also try to include multiple languages into this system to have a greater reach.

REFERENCES

1. Alekh Agarwal and Pushpak Bhattacharyya. Sentiment analysis: A new approach for effective use linguistic knowledge and exploiting similarities in a set of documents to be classified. In Proceedings of the International Conference on Natural Language Processing (ICON), 2005.
2. Shaikh Mostafa Al Masum, Helmut Prendinger, and Mitsuru Ishizuka. SenseNet: A linguistic tool to visualize numerical-valence based sentiment of textual data. In Proceedings of the International Conference on Natural Language Processing (ICON), pages 147–152, 2007. Poster.
3. Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. Just how mad are you? Finding strong and weak opinion clauses. In Proceedings of AAAI, pages 761–769, 2004. Extended version in Computational Intelligence 22(2, Special Issue on Sentiment Analysis):73–99, 2006.
4. Casey Whitelaw, Navendu Garg, and Shlomo Argamon. Using appraisal groups for sentiment analysis. In Proceedings of the ACM SIGIR Conference on Information and Knowledge Management (CIKM), pages 625–631. ACM, 2005.
5. Alexander Pak, Patrick Paroubek from Universit´e de Paris-Sud, Laboratoire LIMSI-CNRS, Bˆatiment 508,F-91405 Orsay Cedex, France, “Twitter as a Corpus for Sentiment Analysis and Opinion Mining.”
6. Bo Pang and Lillian Lee from Yahoo! Research, 701 First Avenue, Sunnyvale, CA 94089, USA, Computer Science Department, Cornell University, Ithaca, NY 14853, USA. “Opinion Mining and Sentiment Analysis.”

7. O'Keefe. T and Koprinska I, "Feature Selection and Weighting in Sentiment Analysis," in Proceeding of 14th Australasian Document Computing Symposium, Dec 2009, Sydney, Australia
8. Pragya Tripathi, Santosh Kr Vishwakarma, and Ajay Lala, "Sentiment Analysis of English Tweet Using Rapidminer," in International Conference on Computational Intelligence and Communication Networks, 2015, pp. 668-672.
9. Martin Sundermeyer, Ralf Schluter, and Hermann Ney, "LSTM Neural Networks for Language Modeling" INTERSPEECH 2012 ISCA's 13th Annual Conference Portland, OR, USA September 9-13, 2012
10. Mangal Singh, Md. Tabrez Nafis, and Neel Mani, "Sentiment Analysis and Similarity Evaluation for Heterogeneous-Domain Product Reviews," in IJCA, vol. 144, no. 2, June 2016.