**Sentiment Analysis of twitter data Using Deep Learning**

**INTRODUCTION**

1.1 Overview

Sentiment analysis (a.k.a opinion mining) is the automated process of identifying and extracting the subjective information that underlies a text. This can be either an opinion, a judgment, or a feeling about a particular topic or subject. The most common type of sentiment analysis is called ‘polarity detection’ and involves classifying a statement as ‘positive’, ‘negative’, or ‘neutral’.

For example, let’s take this sentence: “I don’t find the app useful: it’s really slow and constantly crashing”. A sentiment analysis model would automatically tag this as Negative.

A sub-field of Natural Language Processing (NLP), sentiment analysis has been getting a lot of attention in recent years due to its many exciting business applications.

By analyzing social media posts, product reviews, customer feedback, and NPS responses (among other unstructured data), businesses can understand how their customers feel about their product or service.

1.2 Purpose

Sentiment analysis is particularly useful for social media monitoring because it goes beyond the number of likes or retweets, by providing qualitative insights.

Imagine you just launched a new product feature and notice a sharp increase in mentions on Twitter. Are customers tweeting more because they are delighted with the new feature? Or, are they actually complaining about the feature? Performing sentiment analysis on Twitter data can help you quickly understand the tone of those mentions.

**Literature Survey**

2.1 Existing Problem

Nearly 80% of the world’s digital data is unstructured, and a large portion of that includes social media data.

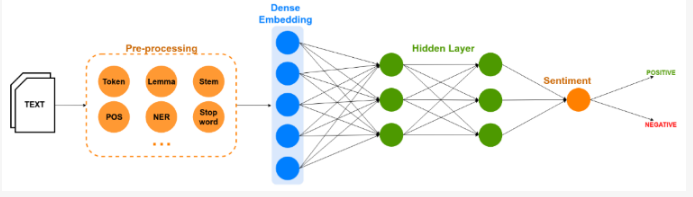
In general sentiment analysis is applied to the Twitter data that can be handled with the Natural Language Processing. The analysis of Twitter data is based upon the classification level to the learning of the words and phrases. The classification of Twitter messages is similar to the analysis of sentiments at sentence level. However, the casual and informal languages used in tweets, the Twitter sentiment analysis is a unique task in microblogging domains. The problem in microblogging domain is how one can work with sentiment analysis techniques on the well-formed data

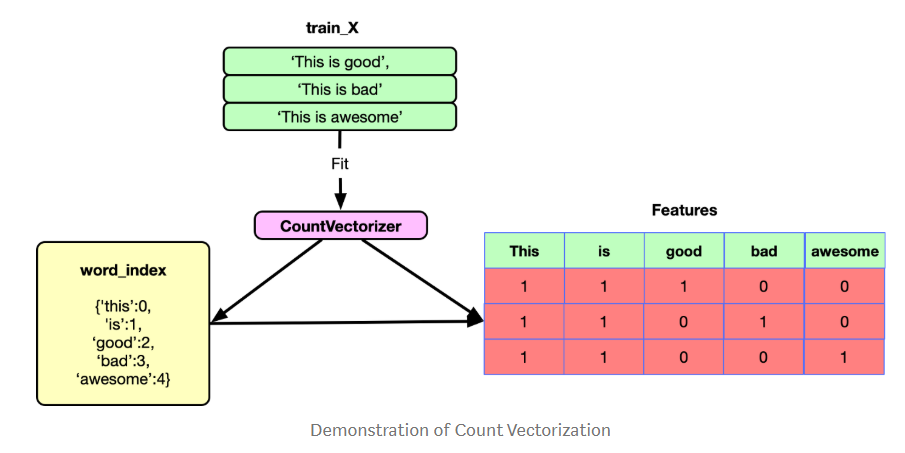
2.2 Proposed Solution

Sentiment analysis tools use machine learning and natural language processing (NLP) to organize unstructured text data automatically. Sentiment analysis algorithms are able to learn from data samples to detect the polarity of Tweets in real-time. All you have to do is train sentiment analysis tools to recognize sentiment in tweets, and they’ll do the rest.

**THEORITICAL ANALYSIS**

3.1 Block diagram



 3.2 Software Desiging

Required Software are Anaconda with jupyter notebook with python 3 version.

Tensorflow with keras libraries are required.

Other required libraries will be available in jupyter notebook.

**EXPERIMENTAL INVESTIGATION**

4.1 Datasets

We train and test our model on the benchmark sets from the KAGGLE Sentiment analysis Compitition.

It’s important that your data is representative of what you’re trying to find out because you’ll use it to:

Train your sentiment analysis model

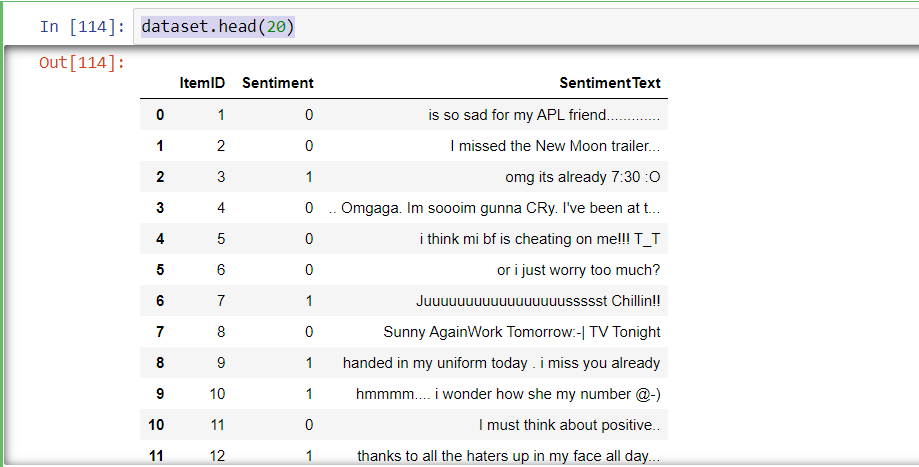
Test how your model performs on Twitter data

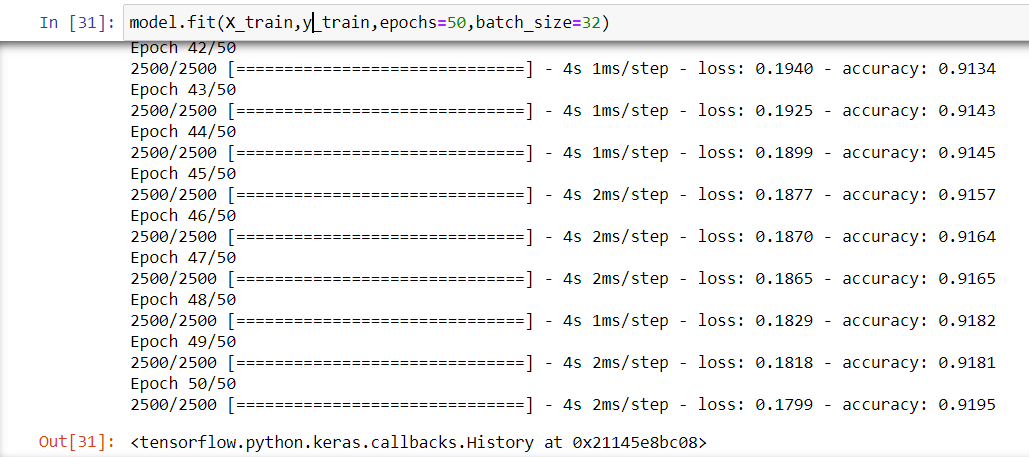
4.2 Experimental Setup

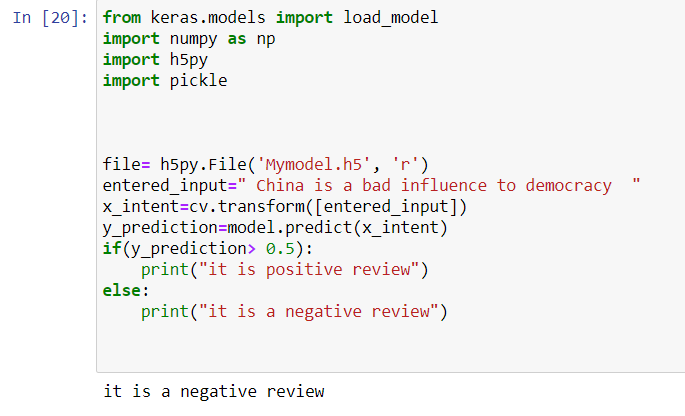
Tweets are retrieved using Twitter API based on a hashtag. The hyperlinks, punctuations, special characters, etc. present in the tweets are removed in the preprocessing steps. The dataset contains 99989 tweets, which are split into training and test datasets in the ratio 0.20. We will use the Sequential constructor to create a model, which will then have layers added to it using the add() method. At this point we have training data and a fully configured neural network to train with said data. All that is left is to pass the data to the model for the training process to commence, a process which is completed by iterating on the training data. Training begins by calling the fit() method. The arguments are batch size as you are using “adam” (bath gradient descent and epochs: no: of times the model should get trained.

Model is to be saved for the future purpose. This saved model ac also be integrated with android application or web application in order to predict something.

**RESULT**







**ADVANTAGES AND DISADVANTAGES**

The main advantage is that they are data driven self-adaptive methods where, they can adjust themselves to the data without any explicit specifications of functional or distributional form. They can approximate any function with arbitrary accuracy since they are universal functional approximates.

One main disadvantage is large complexity of the network structure.

**APPLICATIONS**

Since the Opinion based or feedback based application are more fashionable, now a days, the natural language processing community shows much interest in Sentiment Analysis and Opinion Mining system. The explosion of internet has changed the people’s life style, now they are more expressive on their views and opinions. And this tendency helped the researchers in getting user-generated content easily. The major applications are:

**•**Purchasing Product or Service: While purchasing a product or service, taking right decision is no longer a difficult task. By this technique, people can easily evaluate other’s opinion and experience about any product or service and also he can easily compare the competing brands. Now people don’t want to rely on external consultant. The Opinion mining and sentiment analysis extract people opinion form the huge collection of unstructured content, the internet, and analyze it and then present to them in highly structured and understandable manner.

**•**Quality Improvement in Product or service: By Opinion mining and sentiment analysis the manufactures can collect the critic’s opinion as well as the favorable opinion about their product or service and thereby they can improve the quality of their product or service. They can make use of online product reviews from websites such as Amazon and C|Net , RottenTomatoes.com and IMDb .

**•**Marketing research: The result of sentiment analysis techniques can be utilized in marketing research. By sentiment analysis techniques, the recent trend of consumers about some product or services can be analyzed. Similarly the recent attitude of general public towards some new government policy can also be easily analyzed. These all result can be contributed to collective intelligent research.

**•**Recommendation Systems: By classifying the people’s opinion into positive and negative, the system can say which one should get recommended and which one should not get recommended.

**CONCLUSION AND FUTURE SCOPE**

Applying sentimental analysis to extract the sentiment became an important work for many organizations and even individuals. Sentiment analysis is an emerging field in decision making process and is developing fast. Our project goal is to analyze the sentiments on a topic which are extracted from the Twitter and determine its nature (positive/negative/neutral) of the defined topics. The development of techniques for the document-level sentiment analysis is one of the significant components of this area. Recently, people have started expressing their opinions on the Web that increased the need of analyzing the opinionated online content for various real-world applications. A lot of research is present in literature for detecting sentiment from the text. Still, there is a huge scope of improvement of these existing sentiment analysis models. Existing sentiment analysis models can be improved further with more semantic and commonsense knowledge.

**BIBIOGRAPHY**

Author name: P Shashank Reddy

Title of the work: llSPS\_INT\_3146\_Sentiment Analysis of twitter data Using Deep Learning

NO. of pages: 14

<https://www.kaggle.com/c/twitter-sentiment-analysis2/data>

<https://thesmartbridge.com/documents/spsaimldocs/NLPTExtprep.pdf>

<https://thesmartbridge.com/documents/spsaimldocs/NLPModel.pdf>

<https://www.w3schools.com/bootstrap/bootstrap_forms_inputs.asp>

<https://thesmartbridge.com/documents/spsaimldocs/FlaskML.pdf>

<https://www.researchgate.net/publication/327160507_Sentiment_Analysis_on_Twitter_Data_Using_Machine_Learning_Algorithms_in_Python>

**APPENDIX**

SOURCE CODE:

**Data Collection:**

File descriptions

train.csv - the training set

test.csv - the test set

Data fields

ItemID - id of twit

Sentiment - sentiment

SentimentText - text of the twit

0 - negative

1 - positive

**Text Pre Processing:**

Step 1: Gathering the data and reading the dataset:

1. **import numpy as np**
2. **import csv**
3. **import matplotlib.pyplot as plt**
4. **import pandas as pd**
5. **dataset=pd.read\_csv('train.csv',encoding = "ISO-8859-1")**
6. **dataset.head(20)**

Step 2: Text Cleaning or Pre-processing

We will be using different libraries for the steps that are included in this step.

“Re” is the library which is used to replace the selected special characters with

desired parameter. “NLTK” – Natural language Tool Kit is the library used

for stemming using a special class in the library.

1. **import re**
2. **import nltk**
3. **nltk.download('stopwords')**
4. **from nltk.corpus import stopwords**
5. **from nltk.stem.porter import PorterStemmer**
6. **ps=PorterStemmer()**
7. **data=[]**
8. **for i in range(0,99989):**
9. **review=dataset["SentimentText"][i]**
10. **review=re.sub('[^a-zA-Z]',' ',review)**
11. **review=review.lower()**
12. **review=review.split()**
13. **review=[ps.stem(word) for word in review if not word in set(stopwords.words('english'))]**
14. **review=' '.join(review)**
15. **data.append(review)**

Step 3 Bag of Words:

1. **from sklearn.feature\_extraction.text import CountVectorizer**
2. **cv=CountVectorizer(max\_features=1500)**
3. **X=cv.fit\_transform(data).toarray()**
4. **y=dataset.iloc[:,1].values**

Step 4. Splitting Data into Training and Test set

1. **from sklearn.model\_selection import train\_test\_split**
2. **X\_train, X\_test, y\_train, y\_test=train\_test\_split(X,y,test\_size=0.20,random\_state=0)**

NLP

Importing the libraries

1. **import keras**
2. **from keras.models import Sequential**
3. **from keras.layers import Dense**

Initializing the model

Keras has 2 ways to define a neural network:  Sequential  Function API The Sequential class is used to define a linear initializations of network layers which then, collectively, constitute a model. In our example below, we will use the Sequential constructor to create a model, which will then have layers added to it using the add() method.

1. **model=Sequential()**

Adding Input Layer

This step is to add a dense layer (input layer) where you will be specifying the number of inputs to the neural network, activation function and weights initializer and number of connection to the hidden layer as the arguments. We use add() method to add dense layers.

1. **model.add(Dense(units=20, activation='relu', input\_dim=1500))**

Adding an Output Layer

This step is to add a dense layer (output layer) where you will be specifying the number of classes your dependent variable has, activation function and weight initializer as the arguments. We use add () method to add dense layers. In this layer no need of mentioning input dimensions as we have mentions them in the above layer itself

1. **model.add(Dense(units=1,activation='sigmoid'))**

Configuring the learning process

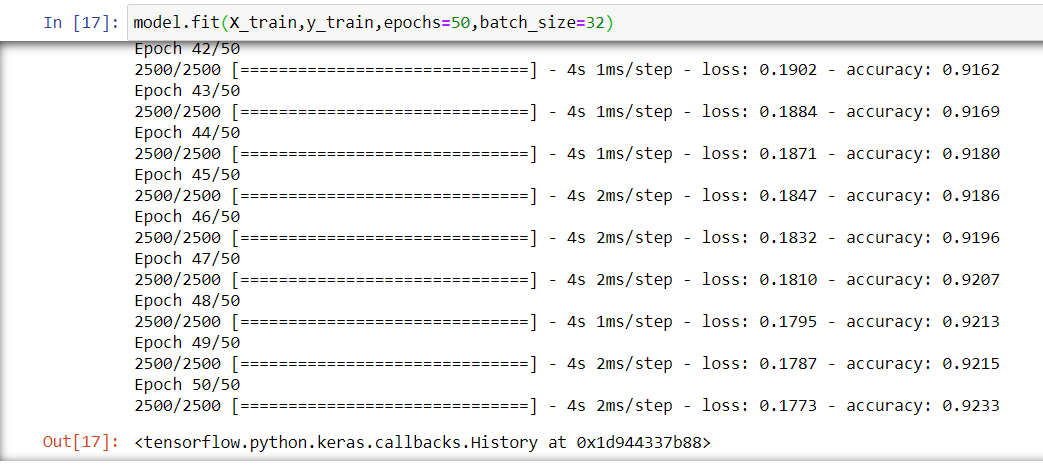
With both the training data defined and model defined, it's time configure the learning process. This is accomplished with a call to the compile() method of the Sequential model class. Compilation requires 3 arguments: an optimizer, a loss function, and a list of metrics.

1. **model.compile(optimizer='adam',loss='binary\_crossentropy',metrics=['accuracy'])**

Training the model

At this point we have training data and a fully configured neural network to train with said data. All that is left is to pass the data to the model for the training process to commence, a process which is completed by iterating on the training data. Training begins by calling the fit() method. The arguments are batch size as you are using “adam” (bath gradient descent and epochs: no: of times the model should get trained.

1. **model.fit(X\_train,y\_train,epochs=50,batch\_size=32)**



Save The Model

Model is to be saved for the future purpose. This saved model ac also be integrated with android application or web application in order to predict something.

1. **model.save('Mymodel.h5')**

Prediction:

The last and final step is to make use of Saved model to do predictions. We use load model class to load the model.

1. **from keras.models import load\_model**
2. **import numpy as np**
3. **import h5py**
4. **file= h5py.File('Mymodel.h5', 'r')**
5. **entered\_input=" bad "**
6. **x\_intent=cv.transform([entered\_input])**
7. **y\_prediction=model.predict(x\_intent)**
8. **if(y\_prediction> 0.5):**
9. **print("it is positive review")**
10. **else:**
11. **print("it is a negative review")**