TWITTER AS A CORPUS FOR SENTIMENT ANALYSIS

BY

MD. ABDUL MUNEEB	2015A3PS0199P
ABHINAV SHARMA	2015B4A70884P
BHARATHA RATNA PULI	2015A7PS0089P
PRASHANT PIYUSH	2015A3PS0248P

Submitted in partial fulfilment of the course

CS F469 - Information Retrieval

Birla Institute of Technology and Science, Pilani

October - November 2017

PROBLEM STATEMENT

Comparison of various machine learning approaches to sentiment analysis of twitter data. Deep learning approaches such as Convolutional Neural Networks (CNN) is compared with other machine learning approaches such as Support Vector Machines (SVM) for tweet classification. We also implement an information retrieval system which takes a query and retrieves the tweets by ranked retrieval and outputs positive and negative tweets related to the query

BACKGROUND

A vital piece of our information gathering conduct has dependably been to discover what other individuals think. Today a great many clients share their perspectives and suppositions on person to person communication locales. For example, Twitter, Reddit and Facebook. Because of the free organization of messages and the measure of clients, Twitter delivers around a million tweets in a month which makes it an incredible corpus for sentiment analysis and supposition mining. Tweets are named positive, negative. This approach is helpful for shoppers who can utilize sentiment analysis to scan for items, for organizations that go for checking general society sentiment of their brands, and for some different applications. As an ever increasing number of clients post about items and administrations they utilize, or express their political and religious perspectives, small scale blogging sites wind up noticeably profitable wellsprings of individuals' feelings and sentiments. Such information can be effectively utilized for advertising or social investigations.

We utilize microblogging and all the more especially Twitter for the accompanying reasons:

• Twitter's group of onlookers fluctuates from customary clients to VIPs, organization delegates, politicians and even nation presidents. Consequently, it is conceivable to gather content posts of clients from various social and interests gatherings.

• Microblogging stages are utilized by various individuals to express their conclusion about various themes, hence it is a significant wellspring of individuals' suppositions.

• Twitter contains a tremendous number of content posts and it develops each day. The gathered corpus can be self-assertively vast.

• Twitter's crowd is spoken to by clients from numerous countries. In spite of the fact that clients from U.S. are winning, it is conceivable to gather information in various dialects.

MOTIVATION

Deep learning is becoming increasingly popular nowadays for text mining as they not only capture lexical and semantic data but also more abstract data. Deep learning architectures have the capability to generalize in non-local and global ways, generating learning patterns and relationships beyond nearest neighbours in the data. On the other hand other machine learning techniques like Support Vector Machines (SVMs) do not capture abstract data but have proven to be sufficiently accurate and reliable. SVM's have shown accuracies upto 85% for sentence classification. Here, we attempt to compare Deep CNN's with SVM in terms of accuracy.

TECHNICAL ISSUES

1. Many of the tweets were very short in length, making their classification very difficult.
2. Hyperparameter tuning of the CNN was difficult.
3. Training the CNN took a lot of CPU time (~5hrs in our case).
4. Since our corpus was large, training the SVM also took a lot of time.

RELATED WORK

Paper 1: Sentiment Analysis on Twitter Data using KNN and SVM.
https://thesai.org/Downloads/Volume8No6/Paper_3-Sentiment_Analysis_on_Twitter_Data_using_KNN_and_SVM.pdf
In this paper sentiment analysis of tweets is performed using the standard machine learning algorithms KNN and SVM. For a small number of tweets KNN performs better but as the number increases SVM performs better. Both SVM and KNN showed accuracies around 80% for sentiment analysis of real time tweets.

Paper 2 : Are Deep Learning methods better for Twitter Sentiment Analysis?
http://www.anlp.jp/proceedings/annual_meeting/2017/pdf_dir/C5-1.pdf
This paper compares the deep learning methods like CNN and RNN with traditional machine learning algorithms like Naive Bayes, KNN and SVM. The conclusions of the paper were that deep learning methods are not necessarily better than traditional learning methods. Their performance depends a lot on their network architecture and also on the size of the training dataset. Traditional machine learning methods like SVM perform better than deep learning methods, especially when the training dataset is small.

SVM showed the max accuracy of around 80% whereas CNN gave an accuracy of about 67%

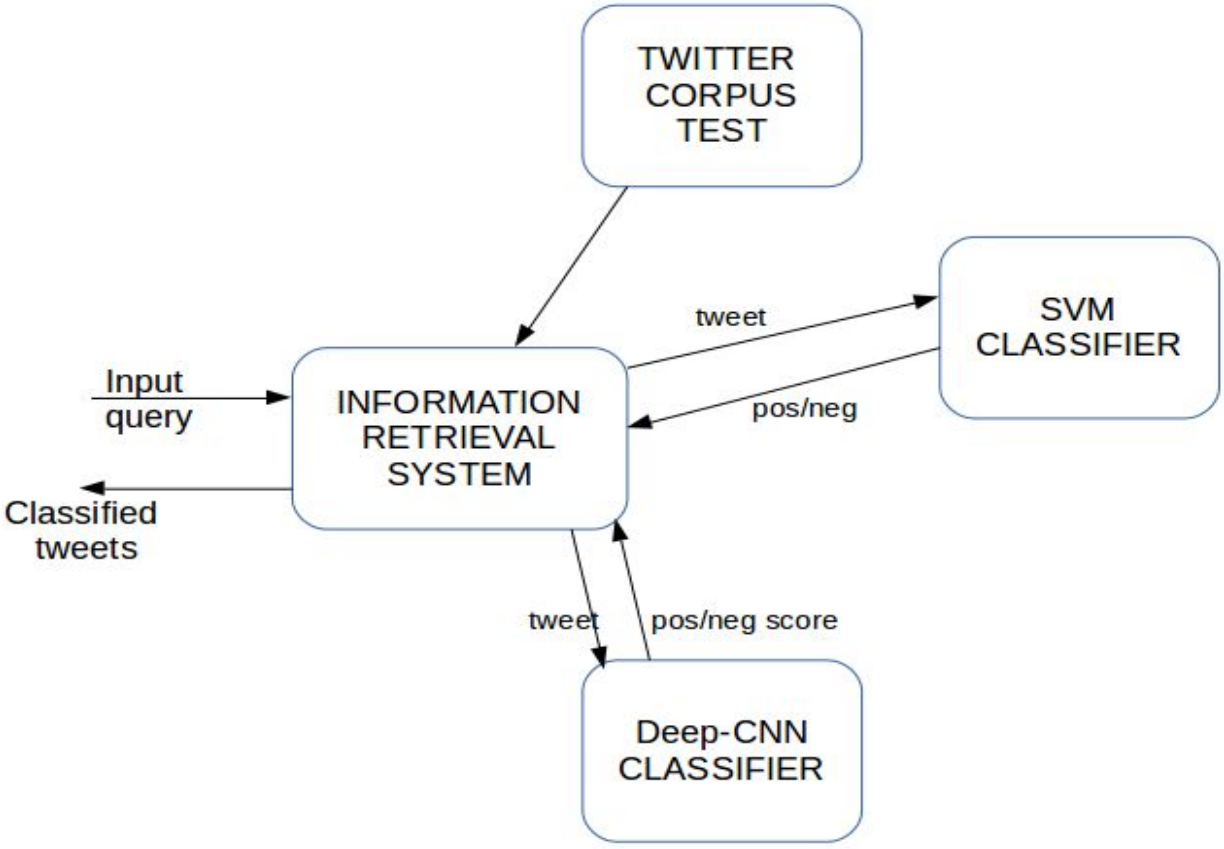
Paper 3: Twitter Sentiment Analysis using Deep Convolutional Neural Network.
https://www.researchgate.net/profile/Gjorgji_Strezoski/publication/279208470_Twitter_Sentiment_Analysis_Using_Deep_Convolutional_Neural_Network/links/55910bf708ae47a3490ef937/Twitter-Sentiment-Analysis-Using-Deep-Convolutional-Neural-Network.pdf
This paper uses Deep ConvNets for Twitter sentiment analysis. It uses a multi-layer neural network and Pretrained GloVe vectors. The size of the dataset used was 4000 and the CNN showed an accuracy of 68%.

MODULES

1.Web Crawling and Data Preprocessing .The collected data is preprocessed(cleansing and representation) and features are extracted. This will be our test dataset.
2.Sentiment Classifier implemented using Support Vector Machine (SVM) and evaluated.
3.Sentiment Classifier is implemented using Deep Convolutional Neural Network (Deep-CNN) and evaluated.
4.An Information retrieval system is implemented using the corpus collected and ranked retrieval of tweets is performed and the results are presented in the form of positive and negative tweets using the classifiers implemented above.

SYSTEM DESCRIPTION

System Block Diagram:



Information Retrieval module:

preproc.py This file provides preprocessing (stemming, filtering, etc) class for twitter corpus data. It parses the corpus into a dictionary.

utils.py provides helper functions that are used throughout the application, in preprocessing, index creation, and query processing.

indexing.py The Indexer class creates the frequency index which calculates tf-idf frequency for ranking and retrieval. The frequency index is used for computing term frequencies and index creation. The frequency index is important to calculate relative term frequency when queries represent it as vectors. The tf-idf is also stored as a dictionary in which the keys are the token words and the values are dictionaries with keys being document ids and values being the tf-idf computed weight. They are saved as txt files for easy retrieval so that the program does not have to spend time recreating the index with every run of the program, and they can simply be loaded into memory via JSON loading functions with every consecutive run.

query.py The Query class is used to retrieve a limited set of documents (i.e. documents containing at least one word from the query), and then rank and return these documents using cosine similarity measures.

system.py System is the main execution class of the Information Retrieval system. In here we initiate the system specifying the location of the twitter messages, the location of the frequency-index (either to be saved or loaded from), and the location of the tf-idf-index (again, either where it should be saved or loaded from).

SVM module:

Sentiment.py This file contains the svm classifier function and other helper functions for vectorization and processing.

Training.py This file contains the functions for preprocessing and training the svm classifier and hyperparameter tuning.

Dataset used is Stanford Sentiment140

http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip

The python library scikit-learn is used and a Support vector machine with linear kernel is used for the classifier.

Deep-CNN module:

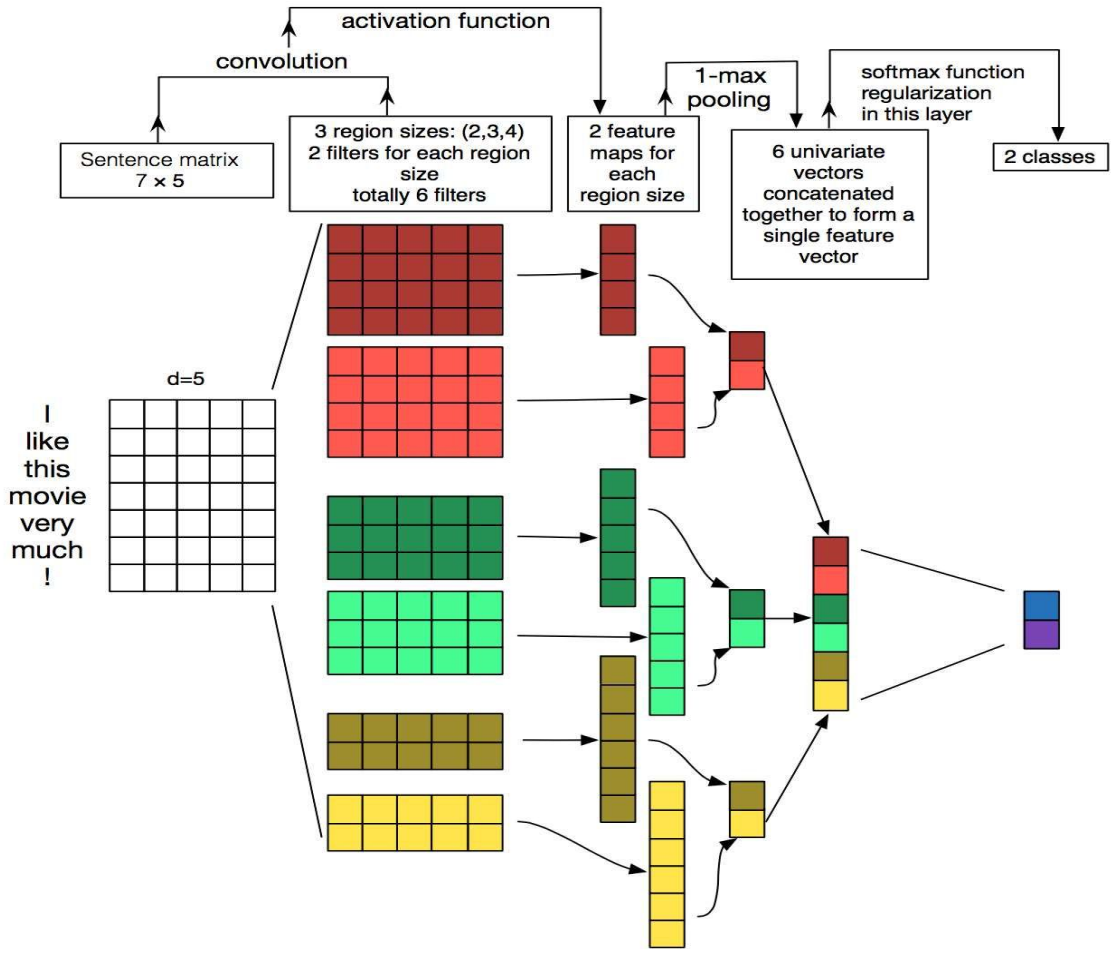
csv_parser.py This file parses the training dataset and divides it into positive and negative sets

vocabuilder.py This file loads the dataset and the word vectors and builds the vocabulary

deep-cnn.py This is the main classifier file which trains and tests the data and also outputs the evaluation metrics.

The neural network is structured as follows:

- Embedding Layer - maps words to vectors (word2vec)
- Convolution Layer - performs convolution and outputs to a pooling layer
- Pooling Layer - Pools the output of the convolution layer
- Concatenation Layer - Concatenates the output of different pooling layers into a single vector
- Dropout Layer - Some random neurons are dropped
- Output Layer - Uses softmax activation function for classification



EXPERIMENTAL RESULTS AND EVALUATION

SVM classifier:

Training the linear SVM classifier was pretty straightforward. We used unigram/bigram indexes and tf-idf vectorisation.

	precision	recall	f1-score

0(negative)	0.82	0.80	0.81
1(positive)	0.81	0.83	0.82

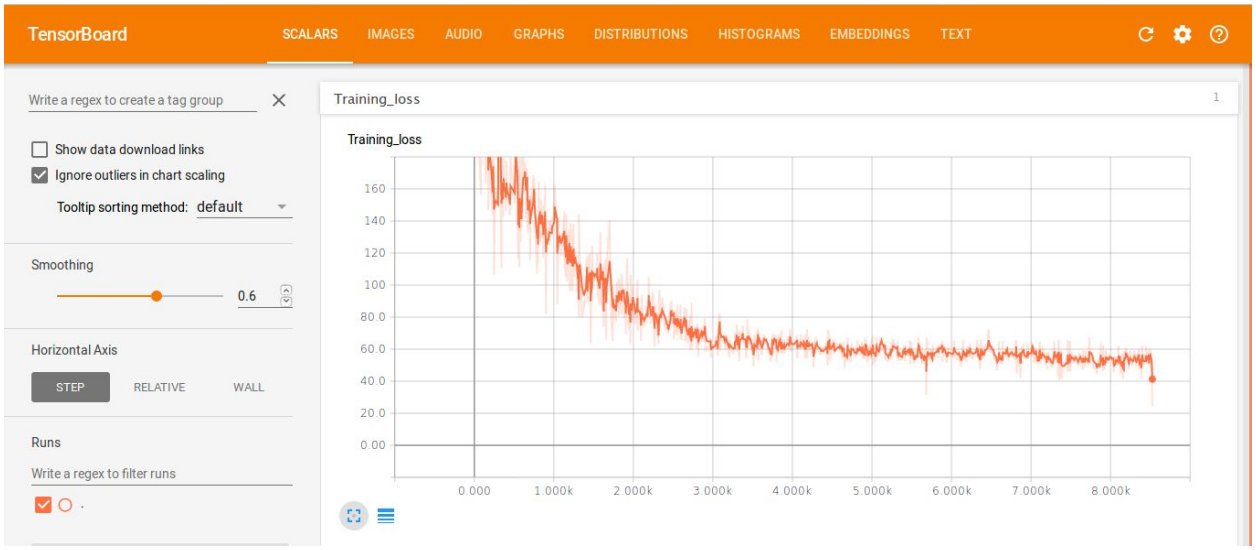
Deep-CNN:

Training CNN was difficult. Our Dataset was huge(~1.5 million tweets) so we had to use only a part of the dataset due to the long computing time involved. We used one third of

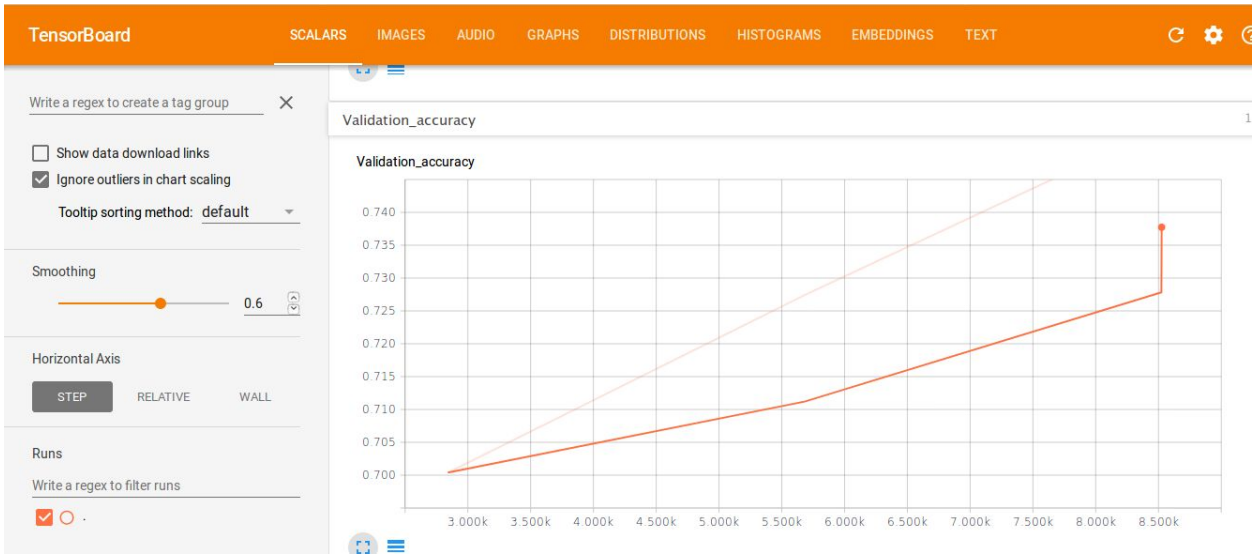
our dataset which came out to be approximately 500,000 tweets. The model took around 5hrs to run and we used pretrained GloVe vectors(400,000) for word embeddings.

Below are the loss and accuracy curves

Loss Curve:



Accuracy Curve:

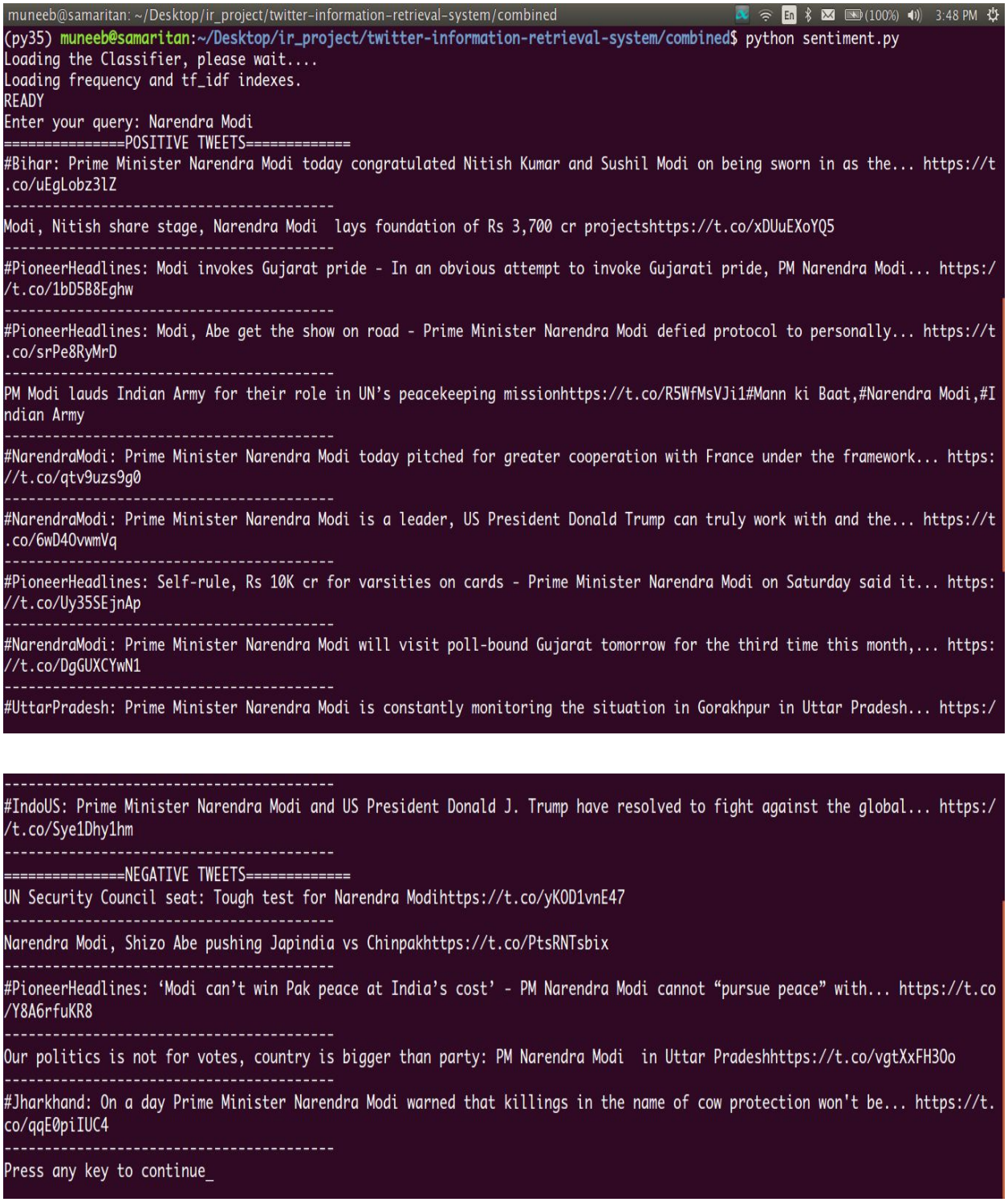


The accuracy came out to be **around 74%**

Information Retrieval System:

The system takes a query from a query and does ranked retrieval of tweets which are stored locally on the system. We made this corpus using crawling and constructed inverted indexes and performed ranked retrieval of tweets based on the query. Here is

an example screenshot. The query is ‘Narendra modi’



CONCLUSION AND FUTURE WORK

Deep learning architectures show a great potential for sentiment classification. These models do not need to be provided with pre-defined features but they can learn sophisticated features from the dataset by themselves and also learn more abstract features . But they have limitations. They require extremely large amount of data and are very computationally expensive to train. If the amount of data is limited, they are unlikely to outperform other machine learning approaches like SVM which are relatively easier to train and tune. We get the accuracies 83% and 73% for SVM and Deep-CNN respectively. Therefore, we conclude that SVM models have rather good fitness to sentiment classification especially when the texts are short and the data is limited.