Dissertation on

Performance Evaluation of Real Time twitter data for Business Analytics using cloud platform

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

UE18CS390A – Capstone Project Phase - 1

Submitted by:

Abhijit Mohanty	**PES1201801293**
Akash K	**PES1201801723**
Gagan Mahesh	**PES1201800793**
TS Yogesh	**PES1201801760**

Under the guidance of

Prof. Guide Name
Designation
PES University

January - May 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

Performance Evaluation of Real Time twitter data for Business Analytics using cloud platform

is a bonafide work carried out by

Abhijit Mohanty	**PES1201801293**
Akash K	**PES1201801723**
Gagan Mahesh	**PES1201800793**
TS Yogesh	**PES1201801760**

in partial fulfilment for the completion of seventh semester Capstone Project Phase - 1 (UE18CS390A) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 6th semester academic requirements in respect of project work.

Signature **<Name of the Guide>** Designation	Signature Dr. Shylaja S S Chairperson	Signature Dr. B K Keshavan Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. _____	_____
2. _____	_____

DECLARATION

We hereby declare that the Capstone Project Phase - 1 entitled **"Title of the project"** has been carried out by us under the guidance of <Prof. Guide Name, Designation> and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1201801293	**Abhijit Mohanty**	**<Signature>**
PES1201801723	**Akash K**	**<Signature>**
PES1201800793	**Gagan Mahesh**	**<Signature>**
PES1201801760	**TS Yogesh**	**<Signature>**

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. <Guide Name>, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE18CS390A - Capstone Project Phase – 1.

I am grateful to the project coordinators, Prof. <Project Coordinators Name>, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

With the advent of the sphere of bioinformatics, there is a new confluence of engineering and biology that has led to remarkable changes in the way researchers approach difficult and time consuming problems. The pharmaceutical industry has highly benefited from this. Research and opened new gates with regards to how we look at drugs and their applications. With this window of opportunity, comes a big challenge: information on these drugs are not computationally friendly. In this work we take advantage of the structural and functional aspects of a drug to generate a drug embedding, an accurate representation which will serve as a gateway to other bio informatic applications like drug discovery, drug target interactions, drug reprofiling and drug repositioning. We employ machine learning techniques to learn the embeddings of the drugs and validate their efficacy by testing against a known biological classification. Additionally, we are aiming to solve an important bioinformatic application which is: early and accurate identification of potential adverse drug reactions (ADRs) for combined medication which is vital for public health. Since most clinical trials focus on a single drug and its therapeutic effects, most drug-drug interaction induced ADRs go unnoticed until the drugs are actually approved. This has been one of the top 10 reasons for death in the United States according to the studies. To solve the same problem we employ various machine learning models to predict drug-drug induced ADRs using the structural and functional characteristics of drugs.

TABLE OF CONTENT

INTRODUCTION **1**

PROBLEM STATEMENT **3**

LITERATURE REVIEW **5**

3.1 Background on Abstractive Text Summarisation 5

3.1.1 Abstractive Text Summarization using Artificial Intelligence 5

3.1.2 Attention is ALL You Need 8

3.2 Background on Emotion detection 10

3.2.1 EmoSense at SemEval-2019 Task 3: Bidirectional LSTM Network for Contextual Emotion Detection in Textual Conversations 10

3.2.2 Emotion Detection in Text: a Review 12

3.2.3 Twitter Sentiment Analysis using Deep Learning Methods 15

3.3 Application of Text Classification and Clustering of Twitter Data for Business Analytics 16

DATA **17**

4.1 Twitter Data 17

4.1.1 Summarisation 17

4.1.2 Sentiment Analysis/Emotion Detection 17

4.2 GoEmotions 18

4.2.1 Training Dataset Split 18

4.2.2 Emotion Classes 18

4.3 Amazon Food review dataset 19

4.3.1 Dataset Summary 19

4.3.2 Attribute Information 19

4.4 CNN Daily Mail 20

4.4.1 Dataset Summary 20

4.4.2 Dataset Version and Features 20

4.4.3 Data Fields 20

4.4.4 Data Splits 20

4.4.5 Average Token Count 20

CHAPTER 5 **21**

SOFTWARE REQUIREMENTS **21**

5.1 Product Perspective 21

5.1.1 Product Features 21

5.1.2 Operating Environment 22
5.1.3 General Constraints, Assumptions and Dependencies 22
5.1.4 Risks 23
5.2 Functional Requirements 23
5.3 External Interface Requirements 25
5.3.1 User Interface 25
5.3.2 Hardware Requirements 25
5.3.3 Software Requirements 25
5.3.4 Communication Interfaces 26
5.4 Non-Functional Requirements 27
5.4.1 Performance Requirement 27
5.4.2 Safety Requirement 27
5.4.3 Security Requirement 27
5.5 Other Requirements 28
SYSTEM DESIGN 29
6.1 SYSTEM ARCHITECTURE 29
6.2 SEQUENCE DIAGRAM/EVENT DIAGRAMS 30
6.2.1 HOME PAGE SEQUENCE DIAGRAM 30
6.2.2 NEWS ARTICLE SEQUENCE DIAGRAM: 31
6.2.3 FOOD REVIEWS SEQUENCE DIAGRAM 32
6.3 DEPLOYMENT DIAGRAM 33
IMPLEMENTATION AND PSEUDO CODE 34
7.1 Twitter Data Extraction 34
7.1.1 Configuring Tweepy 34
7.1.2 Initiating the twitter stream 34
7.1.3 Setting up the listener class 34
7.1.4. Replacing emojis with text 36
7.2 Abstractive Text Summarisation(Fine-Tuning T5) 37
7.2.1 Defining Pre-Trained Model and Tokenizer 37
7.2.1.1 Model Summary 37
7.2.2 Defining Model Parameters 38
7.2.3 Defining Train Step 38
7.2.4 Training and Validation 39

7.2.5 Predicting and Model Demo 39

CONCLUSION OF CAPSTONE PROJECT PHASE-1 41

PLAN OF WORK FOR CAPSTONE PROJECT PHASE-2 42

LIST OF FIGURES

Figure No.	Title	Page No.
Figure 1	System Architecture for the whole platform	26
Figure 2	Attention mechanism Neural Network	8
Figure 3	Home page sequence diagram	27
Figure 4	News Article sequence diagram	28
Figure 5	Food reviews sequence diagram	29
Figure 6	Deployment diagram	30

LIST OF TABLES

Table No.	Title	Page No.

CHAPTER 1

INTRODUCTION

In today's competitive world every organization wants to be ahead of their counterparts in every aspect hence it becomes an important part of the organization to know the emotion of people about the product or the information which they are providing.

Another most important aspect for the quality of the product is the time it takes to convey the message which the product or the article wants to provide. A newspaper organization always wants that it's customers should read through the whole article which they have conveyed but this is only possible if the quality of the headline and the sentiment of the headline is positive towards society.

A new emerging food eatery wants no compromise with the quality of service and wants to have a menu which most of the people like or want to eat. These are small instances which show the importance of people's views on the business model.

It is important to ponder how people are conveying their views on these perspectives. One of the most used platforms is twitter. Twitter is a social media platform where billions of people share their thoughts about many things ranging from news, food, movies, sports etc. Different people respond differently towards a particular product depending on his/her choice. In India people in the southern part like idli and dosa more than chhole bhature which is liked most in northern part. If an eatery wants to grow his business he should be familiar with the people views on the particular food menu he wants to offer.

News headlines and articles are something that everyone discusses and it helps to know what is happening in society, hence it becomes an important business magnate for many stakeholders. But it is important to focus on the quality and content of those news headlines which will help in growth of the news channels. There are cases where the news headlines are provoking and puts a negative impact in the society. No one wants to ruin their morning by reading these negative news.There are cases where news articles are so lengthy that people just skip those news but what is there is some mechanism which will reduce the length but will provide the same information

As discussed in the above scenarios what is the best thing which can be done to provide the views or emotion of the people towards the particular product and how can we shorten the length of the article rather than reading the whole article which may result in not much useful information and to decide whether these articles will give a good impact or not. The solution to this is text summarization and emotion detection which will be discussed in the later paragraphs but before that let's have a look at our project idea.

After going through the multiple use cases, we planned to come up with a product in the form of API which can be used by different business clients to enhance the quality of service they are providing and which will help in growing their business.

We know that in this fast running world all data is now real time and working with real time is a challenging task. Billions of people are using these social media platforms, on an average around 6000 tweets are tweeted on twitter and working with those tweets in a local machine is something which is nearly impossible. To tackle this issue, cloud computing technology helps the most. Using cloud computing the training, testing, hosting and fetching these tweets becomes easier. Training the models and responding to all the requests made by clients requires powerful resources which are provided by cloud providers. Other benefits of using cloud computing consists of data consistency, security, scalability on demand service are some key points which suggests that cloud computing is the need of the hour.

Now the question arises, how to package and deliver everything without any cumbersome work from the client side. The solution to this is REST API's . Api's is a software intermediary that allows two applications to talk to each other . The final product which we will be building will be in the form of API's. Once the client or the organization has these APIs they can access any of the business models they want to use. The API documentation will be provided to the client for understanding all the required details to fetch the results.

Website will also be created to demonstrate how these APIs are used and what is the final output and how effective it is. Real time tweets will be fetched and it will be tested on our platform and the results will be shown.

Now discussing the text summarization and emotion detection, selecting the best model with good accuracy and high rogue score is something which requires a lot of research and enhancements. After selecting those models, performing on the required data sets is also an important task. In this report we will be covering all these details and how we used these ML models in particular on two different datasets which are CNN daily mail and amazon food review dataset and how real time tweets can be fetched and can be tested based on these ML models.

This report is divided into multiple sections ranging from literature review, high level design ,software requirements , implementation etc.

CHAPTER 2

PROBLEM STATEMENT

Consider a hashtag "#idlisambhar" and the tweet such as "*Idli sambhar ...light for the stomach yet so delicious ...can be had anytime anywhere...coconut chutney super healthy"* what can we take inference from this that the particular food item is healthy and can be a good diet who are health conscious. Now if somehow a new eatery gets to know all these reviews and if he can get to know the emotion of the people towards the food then based on it he can grow his/her business. We can find ample number of use case scenarios as mentioned above.

Coming to the solution for scenarios such as mentioned above, how about a user will be provided with a platform, to enter the hashtags or select a newspaper major, and can get these positive or negative reviews, or if some user is interested in news headlines, then the emotion of those. Hence on the similar path we are creating an API which can be used by any business client and they can use it for their websites, mobile apps and many more. For demonstration purposes a website will be built.

Is it an easier task? It does when we listen to the problem statement but it requires many stages using different platforms ranging from third party platforms such as twitter and then fetching those tweets and then testing on pre-built ML models which will be trained on the best datasets. Now let's have a look how these things will be combined and how this platform/product will be created.

The first task is to fetch the tweets either based on hashtags like #idlisambhar #bennedosa and the other category can be of usernames such as CNN, TOI India these are the usernames of some of the best newspaper corporations. After entering the required field there is one challenge, for food reviews we can directly fetch those tweets since the length of tweets of the reviews are not so big and can be easily summarized and emotion can be detected as these statements generally are of smaller length but for news article the tweets does not contain the whole article they contains the url of those article which will be present in their blogs or website hence we have to first scrape the data first and then preprocess it and keep ready for the further use.

Once the required tweets/articles are fetched they have to be given input to the model, we will be using two models one for food review and another for news articles and separate for summarization and separate for emotion detection which will be explained in detail in literature review. The details about the dataset which is chosen for training of these models will be explained in the chapter 4. The fetched tweet may contain emojis which will be

handled in our implementation. The only restriction is that the language which is allowed is english, the reason is straightforward as the ML models which will be used requires to know the grammar and other features which will be cumbersome in different languages such as hindi. Once the tweets/article is given as input there is an option provided to the user either he wants the data to be summarized or he wants the sentiment of the tweet. Only on one click he/she will get to know all the required data which he is interested in.

All these requests responses will be handled by API. Proper documented API documentation will be provided with the API for clear understanding regarding the requests and responses.

All these computations require high computational and powerful resources, for this purpose this platform will be cloud assisted and based on the computational performance we will be performing a performance evaluation on how effective is the real time analysis when it is cloud assisted. ML models will be trained using GPU's which will enhance the platform as well as it will be cost effective. All these measures will be formulated and will be displayed at the end of the computation.

CHAPTER 3

LITERATURE REVIEW

In this chapter, we present the current knowledge of the area and review substantial findings that help shape, inform and reform our study.

3.1 Background on Abstractive Text Summarisation

This section deals with current research and knowledge in this area, findings, and different models which can be used in our product

3.1.1 Abstractive Text Summarization using Artificial Intelligence [1]

This paper gives a brief about different types of summarization and how they differ. They explore Abstractive summarization and compare the performance of seq-seq model and LSTM bidirectional model on Amazon Food reviews and CNN dataset.

Extractive Method: In this method, the predicted final summary involves selecting sentences or phrases from the original text. It selects sentences based on calculated ranks or weights.

Abstractive Method: In this method, the process of summarization involves creating or generating new sentences from the original summary after learning the context. The generated words in the predicted may or may not be there in the original text.
Abstractive methods are in general harder because of semantic and syntactic ambiguity and extractive approaches give better accuracy.

Different methods of sentence extraction

1. Based on features
2. Based on frequency
3. Based on machine learning

Application of Text Summarisation

1. Generating headlines for text such as news articles, reviews
2. Generating notes for students
3. Movie review/preview
4. Product review

Proposed Methodology

1.Recurrent Neural Network

RNN is a type of Deep Neural Network that can remember sequential data by taking previous output as current input at a given timestamp.

2.ROUGE

ROUGE is a metric for evaluating machine predicted summary. It compares the predicted summary and original/reference summary.
ROUGE uses precision and recall

Precision =(*number_of_overlapping_word*s)/(*total_words_in_system_summar*y)
Recall=(*number_of_overlapping_words*) /(*total_words_in_reference_summar*y)
There are different types of ROUGE such as ROUGE-N, ROUGE-S and ROUGE-L

1.Sequence-to-Sequence Model

This model uses the traditional encoder-decoder architecture with attention mechanism.

A. Encoder

Enoder is made up of RNN cells (which can be GRU, LSTM, etc). It encodes the entire input sequence by stepping through time steps into a vector of fixed length known as context vector.
It creates a stack of rnn layers. The outputs of forward and backward layers are combined and given as input to the next layer.

B.Decoder

The decider traverses through different time stamps as it reads from the context vector. In this paper, they use an LSTM decoder while training and Beamsearch decoder during inference.

C. Attention

The drawback of the traditional encoder-decoder architecture is it can't capture the context for long sequences as it has to fit the entire data into a fixed-length context vector.
Attention works by predicting which parts of the input sequence are appropriate to words in the output sequence. It does by giving weights to the input sequences and having a separate neural network for learning these weights. There are 2 types of attention mechanisms: global attention and local attention. In global attention, each element of the contact vector is mapper to all the units in the decoder, whereas in local attention they are mapped to ceratin units in the decoder within a specific window size. The size of the window is empirical. This model provides a value that tells the extent up to which the current input matches with the current output of decoder. Then softmax is used to normalize these scores given below.

The final context vector is obtained by multiplying encoded input with the values /weights generated by the softmax function.

2.Bidirectional LSTM

LSTM can remember sequences better whereas other NN can't. Unidirectional LSTM only considers the input of previous timestamps i.e what it has seen till now, which is not sufficient to get accurate output. So we have a BI-LSTM where one LSTM runs from start-end and the other runs from end-start, at any given timestamp we merge the 2 forward and backward hidden states.

$$\overrightarrow{h^e_t} = \overleftarrow{LSTM}\left(x_t, \overleftarrow{h_{t-1}}\right)$$

$$\overleftarrow{h^e_t} = \overleftarrow{LSTM}\left(x_t, \overleftarrow{h_{t+1}}\right)$$

The decoder uses a softmax layer to model the probability of each target unit into probability distribution which takes the decoder's hidden state and context vector c_t. The concept is based on the Bayes theorem where we maximize the probability of yt given y1,y2….yt-1 i.e to maximize the parameter theta.

$$p(y_t \mid [y_d]_{d \neq t} = \overrightarrow{logp(y_t \mid Y_{[1:t-1]})} + \overleftarrow{logp(y_t \mid Y_{[1:t-1]})}$$

$$\overrightarrow{logp(y_t \mid Y_{[1:t-1]})} = \sum_{t=1}^{T_y} logp(y_y \mid \{y_1, \ldots, y_{t-1}\}, x; \overrightarrow{\theta}$$

$$\overleftarrow{logp(y_t \mid Y_{[t+1:T_y]})} = \sum_{t=1}^{T_y} logp(y_y \mid \{y_{t+1}, \ldots, y_{Ty}\}, x; \overleftarrow{\theta}$$

$$p(y_y \mid \{y_1, \ldots, y_{t-1}\}, x; \overrightarrow{\theta} = g(y_{t-1}, \overrightarrow{h^d_y}, c_t$$

$$p(y_y \mid \{y_{t+1}, \ldots, y_{Ty}\}, x; \overleftarrow{\theta} = g(y_{t+1}, \overleftarrow{h^d_{y,}} c_t)$$

$$e_{ij} = v^T \tanh\left(W^a_h h^a_i + W^e_h h^e_j + b_{attn}\right)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{Tx} e_{ik}}$$

$$c_i = \sum_{j=1}^{Tx} \alpha_{ij} h_j$$

i =Output timestamp ,j=input time stamp

Results

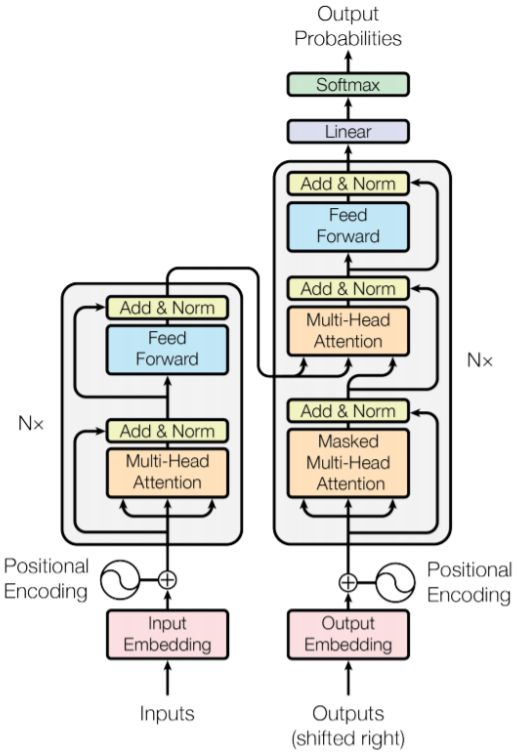
BLEU, ROUGE-1, and ROUGE-2 were used for evaluation.BLEU calculates the precision and ROUGE calculates the recall. Sequence to Sequence model generated the highest BLEU score for Amazon food review. The bi-Directional model for CNN produced the highest ROUGE score.

What we can take from this paper

1. BI-LSTM are able to capture better context by taking into consideration past and future data

2. BI-LSTM model achieved high ROUGE score compared to other model for CNN data,which says this model could be an appropriate candidate for summarising news articles in our product

3.1.2 Attention is ALL You Need[2]



This infamous paper changed the whole NLP talks about the currently most used model transformers. It starts with the explanation of seq-seq models and then describes the transformer architecture, which mainly focuses on attention
Transformer architecture unlike traditional seq-seq architecture avoids recurrence and relies entirely on attention to capture the dependencies between the input and output. It relies entirely on self-attention for computation of input and output representation without recurrence architecture and since they are just activations and weighted sum the computations can be parallelized, unlike RNN.

Transformer Architecture
It consists of an encoder model on one side and a decoder model on another side. Each contains a block of attention and feed-forward neural network which are repeated.

Self Attention
Self-attention is similar to a seq-seq concept where it takes a sequence of input vectors (x1,x2,x3..) and produces a sequence of output vectors (y1,y2,y3…). The output vector as explained is a weighted average of all the input vectors.

Each input is represented in 3 different ways: Query, Key, and value. When the input is compared to other vectors we get Query (yi) and key(yj) and after computation by taking weights we get value. They are represented by 3 different matrices which are learnable, the same encoded input is applied by these matrices. Since they come from the same input we call it "self-attention"

Scaled Dot Product Attention

The 3 values query, key, and value are used for computing attention by taking the dot product of query with all the keys and dividing by the square root of dk(dimension of key) Then we finally apply the softmax function. This attention score tells the alignment of the input with respect to a word at a certain position. They apply "scale" factor to have stabilization over the gradients.After this, the final value is multiplied with the value matrix

$Attention(Q, K, V) = softmax(QK^T)$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Multi-head Attention

Self-attention calculator the attention for the entire sequence and doesn't take order into consideration. To attend to different segments of words, multi-head attention combines different attention heads and divides the word vectors into a fixed number of chunks. Once we get these chunks self-attention is applied to each one of them using query, key, and value matrices. Then these are concatenated to give a single matrix which is taken as input by the feed-forward neural network.

Positional Encoding

Since the attention doesn't take the order into consideration, they use a function that maps the word in a sentence to a vector. This vector is used while learning.The sinusoidal function is used in the paper.

$$PE_{(\text{pos},2i)} = \sin\left(pos/10000^{2i/d_{\text{modes}}}\right)$$

$$PE_{(\text{pos},2i+1)} = \cos\left(pos/10000^{2i/d_{\text{model}}}\right)$$

The Encoder Components

1. Positional Encoding: Converting the input sequence into word embedding after applying positional encoder
2. 6 identical layers each containing:multi-head self-attention and fully connected feed-forward NN as sub layers. This NN is applied to every single token
3. A residual connection is there after each sub-layer, to sum up the outputs which are followed by normalization of the layer and then regularisation

The Decoder Components

The decoder and encoder share some of the components

1. Position encoding
2. 6 identical layers but has 3 sublayers -multi-head attention,encoder-decoder attention, fully connected feed-forward network
3. Residual connection and normalization of layer

Finally, they demonstrate the transformer model on machine translation and other tasks,which shows that the model outperforms all possible models to date.

What we can take from this paper:

1. The transformer doesn't use recurrent architecture and uses only a weighted sum for alignment which makes the computation faster as parallelization can be achieved.
2. The transformer is currently the latest model in NLP and uses in various NLP tasks including summarization which can be used for our product
3. Many pre-trained models are available which can be fine-tuned on custom datasets to match our needs.
4.

3.2 Background on Emotion detection

This section is based on emotion detection

3.2.1 EmoSense at SemEval-2019 Task 3: Bidirectional LSTM Network for Contextual Emotion Detection in Textual Conversations[4]

This paper describes a deep learning model on the Emotion detection in text conversation based on SemEval 2019

They propose a bidirectional LSTM architecture to learn semantic and sentiment feature representation and also to capture user-specific features. They achieved a micro average F1 score of 72.59%

Emotion detection has become an important part of modern-day toxic society, where many people get affected by toxic comments online. So SemEval 2019 focused mainly on contextual emotion detection. So a user utterance is given in text along with 2 turns of contextual conversation. Based on this the emotion of the next user utterance should be classified as either "happy," sad"," angry" or "others"

Model Architecture and System Description

RNN is a type of neural network which are able to learn the sequence in the input compared to traditional neural networks which cannot capture sequence in the data. So they take an output of the previous timestamp as input for the current timestamp along with the current input.LSTM is an improvement over RNN to solve the vanishing gradient problem. But LSTM though are good at only predicting data from what they have seen till now (or till the currently processed inputs) which might sometimes not give correct output based on context. So BI-LSTM are used which capture the sequential data from both directions. One LSTM goes from start to end and another from end to start, at any given timestamp we concatenate the hidden state to get the data captured from both directions.

Proposed Architecture
The proposed architecture consisted of an embedding unit and two bidirectional LSTM units of dimension 64. One LSTM is used to process the utterance of 1st user (1st turn and 3rd turn in the conversation), and another LSTM to process the 2nd user (the 2nd turn). These LSTM cells learn semantic, sentiment feature representation and user-specific conversation features for more accurate classification of emotions. At first, each user utterance is passed onto its respective BI-LSTM layer after using pre-trained word embedding on it. Three feature maps are created for 3 turns, they are concatenated into a flattened feature vector. Then they are passed to a fully connected hidden layer having a dimension of 30, this hidden layer analyses the interactions between the input vectors given. Finally, the output is passed onto a softmax activation function before predicting the final emotion class label. In order to reduce overfitting, they added regularisation layers with Gaussian noise after each embedding layer and dropout layers were added to each LSTM unit with a probability of 0.2 and before the fully connected hidden layer with a probability of 0.1

Training and dataset
The dataset consisted of 30,160 human labeled tweets. About 5000 samples were from each of the classes “angry”, “sad”, “happy” and around 15000 for “other” classes. They were also provided with dev and test datasets which have a real-live distribution of tweets that is -each emotional class having 4%rest of them for “others” class. On top of this data provided they further collected 900k tweets with around 300k for each emotion and sampled it. The metric used for evaluation is the F1 score for the classes ‘sad’,’ happy’ and angry’.

Pre-processing
Preprocessing was done by Ekphrasis tool. This tool is also used for processing emojis in tweets.
This tool can be used for spell correction, normalization, and annotation of special tags, word normalization, word segmentation, etc.
It also provides a tokenizer which has the feature for identifying various emoticons, emojis, complicated expressions such as times, dates, etc.

Unsupervised learning
For word embedding which is the major step in pre-processing, they tried it out using GloVe, Word2Vec, and FastText.Word2Vec being supervised method and glove unsupervised. They represent the sentences/words in vectors such that the contextual information is preserved in the numerical representation of vectors.Word2Vec tries to predict the word, with the goal of minimizing the loss function. After using word embedding they are passed to train the LSMT model, they compared different embedding models using cross-validation.DataStories embedding gave the highest F1 score

Distant Pre-Training
They fine-tuned the word embedding while training. During the first epoch, only the LSTM was trained and the word embedding weights were frozen. And for the later five epochs, both the embedding and LSTM were trained.

Supervised Training
For training, they used Adam Optimiser and a learning rate of 0.01 initially. In the end, they achieved better accuracy with frozen word embeddings and training for 15 epochs.

Experiments and Results
They carry out the experiments in various angles such as with a number of cells, regularization parameters, activation functions. The proposed architecture model achieved an F1 score of 72.59% which was above the baseline for competition 58.68%.

3.2.2 Emotion Detection in Text: a Review [4]

This paper deals with the definition of emotion, the application of emotions in the real world, the difficulty in implementing an accurate model for multi-class classification, and also explains various non-deep learning-based supervised approaches and unsupervised approaches for predicting emotions.
Sentiment analysis is a well-established field in NLP and there are various approaches in implementing a highly accurate model. But the amount of useful information that can be interpreted from just positive and negative labels is limited. And most of the positive and negative sentiments are not created equally such as Fear and Anger are both negative sentiments.

Use cases of emotion Detection

- In marketing emotion detection is used to analyze consumers/customers reaction to products and services and get feedback on which product should be improved
  To satisfy customers

- Emotion detection is also used in recommendation systems to improved users' recommendations based on emotion the current or past history of items.
- Emotion can be used to catch the sentiment of a story article posted online to make the story deeper
- Profiling of writers/authors by processing the emotions in text by them

The complexity of Expressing Emotions in Language

Emotion detection relies heavily on context and is complex. Complexity can be attributed to multiple factors such as sensitivity to different contextual and personal circumstances and another fact that these texts usually contain a mix of different emotions rather than a single emotion so context is very important.

Another issue is implicit emotion expression that is sentences that have emotion in them without explicit emotion words such as(cry, fight, etc) for eg a sentence such as 'go on first date' as emotion embedded in it without any explicit emotion converting word.

Resources for Emotion Detection

1. Being a new field to NLP not many datasets were available in the past but currently, new quality datasets have arisen. Some of the popularly used datasets are
2. SemEval which consists of 2 users having a conversation and the goal is to predict the emotion of utterance of a user in the conversation. The reason 2 users used is to capture the context of the conversation.
3. Swiss Center for Affective Sciences -(SCA)=It consists of around 7600 examples which are responses of 3000 people for emotions belonging to classes ('anger',' fear',' joy',' sadness', disgust',' shame',' guilt')
4. EmotiNet is created based on the idea that a single sentence is not enough and who situation is required for detecting the emotion. They created a knowledge base that had many actions and emotions associated with them. They took examples from the ISEAR database and clustered the example based on the class of emotion
5. Another more easily available dataset is tweets with hashtags as labels and also using emoticons in the tweets for capturing the context

Word Embedding

Word embedding is used to give semantic vector representation of the sentences and words. The idea is based on the fact that words which co-occur frequently in large text corpus are semantically similar. Word is represented by n-dimensional vector space such that the distance between the vectors represents the semantic similarity. Different word embedding such as Word2Vec, Glove is present.Word2Vec is created using supervised approaches

whereas glove is by unsupervised approaches. These embeddings increase the accuracy of the model in various NLP tasks.

Supervised Approaches in Emotion detection
In one of the approaches, tweets from Twitter were used as a dataset having hashtags and emoticons as a class label and as contextual info. They achieved accuracy between 75% to 91% on the manually labeled dataset. Purver and Battersby (2012) using the same dataset achieved 82% using SVM for the happy class using 10-fold cross-validation and 67% for the entire dataset without cross-validation but for classifying on general another dataset they achieved an accuracy of 13% to 76
Unsupervised Approaches
For unsupervised methods, they used various dimensionality reduction methods Latent Semantic Analysis -(LSA), Non-negative Matrix Factorization -(NMF), and Probabilistic Latent Semantic Analysis (PLSA) while considering Wordnet affect as a lexicon. They used the same metrics for evaluation and achieved more or less the same score as supervised approaches.

What we can take from this paper:
Complex Nature of Emotion Expression:

1. Expression of emotion requires context and the context can change many times in a sentence. And each sentence can contain more than one emotion
2. Presence of implicit emotion in a sentence

Shortage of Quality Data

1. In this paper, they described various models and in various approaches, accuracy was affected because of quality data. Though a huge amount of data is available annotated data is required.
2. Because of the diversity of data, the model trained on tweets can't be used on news comments

The inefficiency of Current Models

1. When it comes to multi-class classification non-deep learning models don't achieve good accuracy. If they do achieve high accuracy in a class they fall short in another class.
2. More complex models having neural architecture are required for better accuracy.

3.2.3 Twitter Sentiment Analysis using Deep Learning Methods

This paper explains how to use deep learning, one of the machine learning techniques so as to process huge amounts of unstructured data from Social Network Service (SNS) such as twitter data that includes posts, tweets, comments, retweets, etc. This data is actually very useful for performing tasks like emotion detection.
Twitter can be used to detect things like what is the trending topic under discussion, and sentiment analysis of tweets related to Social Problem can be helpful for government departments such as the social and health ministry and other NGOs. These organisations can implement certain policies so as to reduce these problems.
The sentiment analysis' job is to distinguish text sentiment polarity. We can consider sentiment analysis as a classification problem. This paper explains about classifying the text into either positive or negative sentiment.

Model and Research Methods
This system's design involves the following steps: Downloading twitter data using twitter api, cleaning and preprocessing twitter data, dividing the tweets based on their language, deep neural network training, and testing of sentiment analysis using trained neural network.

Text Mining
Text mining or text analysis is a technology based on AI that uses NLP so as to transform unstructured data into structured and normalised data to perform analytics. The three steps of text mining includes: Information retrieval, Information extraction and Data mining. This paper has used only information retrieval and information extraction.

Text Cleaning and Preprocessing
This is to clean the parts of the tweets that make the analysis part difficult. White spaces, punctuations, etc are removed as these don't help in sentiment analysis. Words like 'the', 'is', etc are removed as well. Other cleaning steps include lemmatization, converting all the words to lowercase and removing numbers.

Deep Learning Methods and Deep Neural Network
Deep learning is a machine learning technique that includes networks which can learn unsupervised from unstructured data. Deep learning network's specification of this paper includes: Feedforward NN, three hidden layers, Relu and sigmoid function's usage, 100 neurons input, MSU(Mean Square Unit) and stochastic gradient descent's usage.
Words are filtered by the first hidden layer, second hidden layer performs filtering based on sentence and finally the third layer performs filtering based on the word's popularity in the online dictionary.
We can use gradient descent to minimise any objective of the form J(Q) where $Q \in RD$. We can achieve this by updating the parameters in the direction opposite to that of the objective function's gradient with respect to the parameters.

Preliminary Result and Conclusion

In this paper for the purpose of training and testing, they have used 1000 dataset each of positive and negative and 4000 is the total. The learning rate used for this experiment is 0.1 and 0.001 and is trained with 100 epochs. The network of this experiment is created using Tensorflow.

The accuracy of the training model goes up and converges at 55%. The model has achieved the accuracy of 67.45% and 52.60% on training and testing dataset respectively.

3.3 Application of Text Classification and Clustering of Twitter Data for Business Analytics

This paper demonstrates the text analysis process in reviewing the public opinion of customers towards a certain brand and presents hidden knowledge (e.g. customer and business insights).

A Twitter API is used to collect twitter corpus and feed it to a Binary Tree classifier that will discover the polarity lexicon of English tweets, whether positive or negative.

A k-means clustering technique is used to group together similar words in tweets in order to discover certain business value.

In the paper a difference was shown that sentiment analysis can be classified into two categories, machine learning approach and lexicon-based approach.

A Decision tree model is used for training and testing the dataset after removal of stop words, words having word length<3 and then a tree description was suggested.Later K-means classification was performed on the same dataset to group together similar words in tweets in order to discover certain business value. Rapidminer, a data-driven data mining tool that discovers patterns in large collections of text, was used for text classification.

The key points to take from this paper:

1. This paper provided an introduction and rationale behind the value of text analytics of Twitter data to businesses in gaining customer views on products and services, and brand.
2. The paper attempts to discuss the technical and business perspectives of text mining analysis of Twitter data and recommends appropriate future opportunities in developing this emerging field.
3. The research was unable to take into the consideration of scaling as a result better algorithms such as SVM could have been considered.

CHAPTER 4

DATA

4.1 Twitter Data

The first step involved in the entire product lifecycle is the data extraction from the Twitter platform. Firstly we do not extract any tweet that is not of the English language. There are different categories of extraction that our product needs:-

4.1.1 Summarisation

1) Extract only the text part of the tweet based on a hashtag given by the user for 'Food reviews' summarisation'. In this part, the user gives a custom hashtag, based on this hashtag we do a search using the standard tweepy framework. We then remove any emojis encountered during the search (based on the given hashtag). Once we get the text, we directly transport it to the summarisation model for food reviews which then gives a summarised output. A minimum length for the tweets extracted can be set.
2) In the second category of extraction (mainly used for news article analysis), we divide it into two steps:-
    a) We first search tweets from a given hashtag or user account (in this case, it involves news corporations such as TOI, CNN ,etc.).
    b) Once we get the tweets from the above sources, we filter out the tweets that do not contain a hyperlink to a news article.
    c) We now have the tweets with hyperlinks to news articles, so undoubtedly the next step is to extract these hyperlinks.
    d) The extracted hyperlinks are now used by a web crawler to extract all textual data from the web pages that the hyperlinks point to.
    e) The data extracted from the web pages is now sent to the "news article" summarisation model.

4.1.2 Sentiment Analysis/Emotion Detection

1) We can use python's demoji module to decode/convert emoji into text that well describes the emoji that is recognised.
2) It is used to accurately remove and replace emojis in text strings
3) Emoticons such as :) can be easily replaced with text that best describes each emoticon.
4) These text descriptions can then be used by the appropriate models to process tweets that contain emojis and provide accurate expected results.
5) One expected challenge is that, emoji list itself is frequently updated and changed.
6) But we can tackle this challenge by periodically updating the local cache by calling emoji.download_codes() function which is provided by the demoji module.

4.2 GoEmotions

GoEmotions is a dataset by google which consists of around 58k examples taken from comments of reddit social platform.Unlike other datasets it has 27 curated emotion categories.

Features	Value
Number of Labels	27 + (neutral emotion class)
Number of Examples	58,000
Maximum Sequence Length	30

4.2.1 Training Dataset Split

Dataset Split	Number of Instances in Split
Train	43,410
Validation	5,427
Test	5,426

4.2.2 Emotion Classes

Dataset :GoEmotions	Value
Classes	Amusement,annoyance,anger,admiration ,excitement,disapproval,nervousness,joy ,approval, caring, confusion, curiosity, desire, disappointment,disgust, embarrassment, excitement, fear, gratitude, grief,love, optimism, pride, realization, relief, remorse, sadness, surprise.

4.3 Amazon Food review dataset

4.3.1 Dataset Summary

Amazon food review dataset has food reviews spanning from amazon spanning from Oct 1999-Oct 2012.The dataset has various attributes such as Helpfulness Denominator which indicates the number of users who found the review helpful,rating of the product and mainly summary which will be used in our product for summarisation.

Features	Value
Number of Reviews	568,454
Number of Unique Users	256,049
Total Number of Products	74,258
Number of Columns/fields	10

4.3.2 Attribute Information

1.Id : Row Id

2.ProductId:Unique number/identifier for the product

3.UserId:Unique number/Identifier for the user

4.ProfileName:Name of the user

5.HelpfullnessNumerator:No of users who found the review helpful

6HelpfullnessDenominator:No of users who indicated if they found the review helpful or not helpful

7.Score :Rating of the product between 1-5

8.Summary:Summary of the original review

9.Text :Original text written by the user.

4.4 CNN Daily Mail

4.4.1 Dataset Summary

CNN Daily mail is a dataset which has over 300k examples of unique news articles which are written by journalists at Daily Mail and CNN.There are different versions of dataset which are created for different NLP tasks such as summarisation ,Question Answering system etc

4.4.2 Dataset Version and Features

Among various versions ,version 2.0.0 and 3.0.0 are used for extractive and abstractive summarisation.The model is evaluated using ROUGE score by comparing the machine generated summary and the original article in the dataset.We achieved a ROUGE score of 40.

4.4.3 Data Fields

id:A unique number associated with every story

article: The main body of the article as a string

highlights:The summary written of the article by the author

4.4.4 Data Splits

For training we used the version 3.0.0

Dataset Split	Number of Instances in Split
Train	287,113
Validation	13,368
Test	11,490

4.4.5 Average Token Count

Feature	Mean Token Count
Article	781
Highlights	56

CHAPTER 5

SOFTWARE REQUIREMENTS

5.1 Product Perspective

The product will mainly be distributed to the users in the form of APIs which they can then use to build their websites (Frontend). All the computation will be done on the cloud, hence, removing the need for setting up the environment to use the product in the local device and also increasing efficiency in terms of speed and resource usage.

For the testing purpose a sample website will be created and how to use APIs will be demonstrated. An API documentation will also be provided to the end user.

The User has to enter the required hashtags or select the appropriate news corporation and will have to select either emotion detection or summarization. Based on this, if the tweets have a length more than a given threshold, the data will be summarized and then emotion detection will be performed on that hence the user will be able to make a judgement based on the review for the given product.

5.1.1 Product Features

The following are the features of our product:-

1. Allow users to worry only about the front end by providing REST APIs that do all the work.
2. Remove the need for having to set up the system by making sure that all the computation is done on the cloud. This is done by storing the models and the required training datasets in the s3 buckets of AWS or the corresponding features of any other cloud service provider.
3. Help users to use the product to analyse which news articles or food items can be marketable by looking into news corporations and trending hashtags respectively, and their associated user reviews.

4. Ensure high accuracy in the model to enable efficient analysis to aid the user in making effective business decisions.
5. Allow the user to either choose food review analysis or news articles analysis or both through APIs.
6. Analyse data real time, hence, removing the need to store the data locally.

5.1.2 Operating Environment

The system will handle all the computations that include training and deploying the model in the cloud. Mainly, the product has been tentatively planned to be deployed in AWS beanstalk. The operating system of the EC2 instance being used will be ubuntu. The software components being used will be postman [for testing the APIs], Visual Studio code for code development, Any modern browser with html5 support. The training dataset will be stored in s3 buckets in AWS. The endpoint will be used to develop the REST APIs that will be shared amongst the users. Twitter will be used as the social media tool to analyse public data for news articles and food reviews. Tweepy (the standard twitter framework) will be used to extract data from Twitter. Web crawlers will be used to extract data of news articles whose hyperlinks will be extracted from tweets. This will save us time from having to crawl through the entire internet for a particular trending topic as it will all be extracted from the tweets of the news corporations (for news article analysis).

5.1.3 General Constraints, Assumptions and Dependencies

1. Users are required to have good internet support as the APIs will use the resources on the cloud for all computations.
2. Modern web browsers that support html5 are a must.
3. Users' data extracted shouldn't be shared amongst the general public in any way. All data extracted will be used for internal purposes only.
4. The end software will be easy to use and there is minimal criticality of the application usage. The user will be able to use the software without any cumbersome work.
5. Software should be fault tolerant and reliable due to the usage of cloud service providers.

5.1.4 Risks

1) Users may not have a persistent internet connection.

2) Risk of leakage of data being extracted from twitter posted by other users.

3) Users may not be using a new web browser.

4) Using GET or any other vulnerable protocol in transfer of sensitive data (such as data extracted from tweets) is risky.

5.2 Functional Requirements

1. **USAGE OF REST APIs:**
   - 1.1. Users should be able to use the REST APIs easily.
   - 1.2. REST APIs should be designed such that it can be called easily.
   - 1.3. The responses obtained from calling the APIs should be appropriate. For ex: a server error should give a response of "500 internal server error".
2. **CLOUD BASED OPERATIONS:-**
   - 2.1. The endpoint hosted on the cloud should work throughout the lifecycle of the application.
   - 2.2. All the computations regarding training and testing the model should be done using the cloud's resources.
   - 2.3. The training dataset should be stored in the s3 buckets in AWS or any other corresponding storage features provided by the cloud provider.
   - 2.4. Any output stored as CSV will not be stored locally. It should only be stored in s3 buckets or any other corresponding storage features provided by the cloud provider (only if required).
   - 2.5. Effective data hiding to prevent leakage of user's data (extracted tweets).
3. **TWEET BASED EXTRACTION:-**
   - 3.1. Extract data using trending hashtags related to food reviews and news articles.
   - 3.2. Users should give the option to filter based on a trending hash tag. for ex: an administrator from a food corporation can give a hashtag as 'biryani'. This allows the api to extract data (reviews given by many users) based on the title of the hashtag.

3.3. Tweets extracted are converted into texts.

3.4. All languages barring English are ignored.

3.5. Emojis are also supplied to the model.

3.6. We can use python's 'demoji' module to decode/convert emoji into text that describes the emoji that is recognised. It is used to accurately remove and replace emojis in text strings.

3.7. Emoticons such as ":)" can be easily replaced with text that best describes each emoticon. These text descriptions can then be used by the appropriate models to process tweets that contain emojis and provide accurate expected results.

3.8. One expected challenge is that, emoji list itself is frequently updated and changed. But we can tackle this challenge by periodically updating the local cache by calling emoji.download_codes() function which is provided by the emoji module.

3.8.1. Example: text= "'sometimes I'm 😌, and sometimes I'm 😕. But most of the times I'm 😩 and 🥱'".

3.8.2. output of demoji.findall(text):

3.8.3. '😌': 'relieved face'

3.8.4. '😕': 'confused face'

3.8.5. '😩': 'tired face'

3.8.6. '🥱': 'yawning face'

5.3 External Interface Requirements

5.3.1 User Interface

1. The main point of contact between the user and the in-house developed training models will solely be through REST APIs. A website will be developed to demonstrate the practicality of this API.
2. The Rest APIs will be developed using Flask. The user can use the REST APIs to give out a hashtag they want to filter upon. The filtered data will then be submitted to the analysis model which will give out the respective analysis (sentiment, toxicity,

etc). The given analysis will then be converted to JSON which will then be supplied to the user as responses complying to the standard REST architecture rules, through the APIs.

3. The timing between the API call and the response should not be more than 15-25 seconds.
4. Any error during the API call will be given an appropriate error message that complies with the standard REST architecture.

5.3.2 Hardware Requirements

The protocols that will be used will comply with the REST architecture. All the computation regarding training and testing the model will be done solely on the cloud. Cloud requirements are EC2 instances, beanstalk environment, S3 buckets linked to the beanstalk environment, RDS instances linked to the same and periodic snapshots to avoid losing any data. The website can be created with the help of aws amplify which allows vendors to host a react website.

The user should only be responsible in specifying the hashtags and selecting major news article corporations (this option will be displayed by the application to the users) to do analysis on and hence should support a modern web browser and persistent internet connection.

5.3.3 Software Requirements

REST APIs:-

Architecture: REST

Protocols used: HTML

Version: 1.1

Tools and libraries: Flask, Python, HTML, JSON

Testing software: POSTMAN

Web browser: any modern web browser.

Description: This is the main contact point between the user and the product. Rest APIs will be called by the user either in the web browser or postman, the appropriate response message with the necessary data will be given to the user.

TWITTER DATA EXTRACTION:

Platform: Twitter

API: Tweepy

Tools and libraries: Tweepy, sqlalchemy, sqlite, JSON, Python,demoji

Testing software: Microsoft Excel

Access tokens: Twitter developer account [consumer key, consumer Secret code, access token, access secret token].

Description: This handles tweets extraction based on trending hashtags related to a certain topic and converts into data suitable for the training model. Emojis are handled appropriately and all languages other than english are ignored/dropped. The output generated after preprocessing data (tweets) should be stored as a csv file for visual verification by the developer.

5.3.4 Communication Interfaces

Communication between the client and server should utilize a REST-compliant web service and must be served over HTTP Secure (HTTPS). All data generated must be stored in a CSV file in the s3 bucket. This should later be converted into JSON format which will be sent as a response to an API call (if successful) to the user. If the API call is not successful, an appropriate error message will be sent to the user which complies according to the REST architecture. The port used for communication will be port 80 [http] (for testing) and port 443 [https] (for users).

5.4 Non-Functional Requirements

5.4.1 Performance Requirement

The software must be light enough to run on low power devices such as cheap mobile phones, tablets, etc. This is to ensure that people from every walk of life can access it without being restricted by their situation or financial conditions. The content must render correctly and must be readable even on smaller screens. The backend of the website must be scalable enough to provide the pages (home screen, emotions and summarized data) quickly (within a few fractions of a second) even under high load. The Emotion detection functionality must scale well, and the performance must not degrade even when there are thousands of users. To

be specific, the summarized and emotion detection results must be available within a few seconds of submitting the query.

5.4.2 Safety Requirement

As content is user-submitted, there is a possibility of potentially harmful information being submitted and accepted. To prevent this, the standards of tweets will be checked so that there will not be any compromise in the quality of content. This will be governed by the accurate model which will be used.

As the Website (software) will be hosted online there is a possibility of database failure. To handle this issue there will be a mechanism to migrate the database so that the repositories are safe during catastrophic events.

5.4.3 Security Requirement

The user log details will be maintained in the cloud server. Two way authentication can be applied either with password and gmail or password and phone. A privacy concern that users could have is that the pages that they view through the product can be used to profile them and can be used to create an online advertising target out of them. This can be prevented by avoiding any kind of storage (such as cookies) for readers i.e., the pages that they view must not be tracked in any way.

5.5 Other Requirements

Software Quality Attributes

1. Robustness : The user can access the Web page(software) without any delay .
2. Correctness: The tweets determine the correctness of data, more tweets gives a more than eases the work of storing the files and images and it’s easy to maintain.
3. Flexibility : The software is easy to use by any user.
4. Availability: The software will be accessible and from anywhere and anytime.
5. Simplicity: The ease of summarization and emotion detection is one of the characteristics the team will focus on. There will be no compromise with the quality of content.
6. Usability: Clean and intuitive user interface.

Appendix A: Definitions, Acronyms and Abbreviations

HTTP: HyperText Transfer Protocol

Qos: Quality of Service

AWS : Amazon web services

REST: Representational state transfer

API: Application programming interface

Appendix B: References

Tweepy: https://docs.tweepy.org/en/latest/

AWS: https://aws.amazon.com/

API Testing software: https://www.postman.com/

CHAPTER 6

SYSTEM DESIGN

6.1 SYSTEM ARCHITECTURE

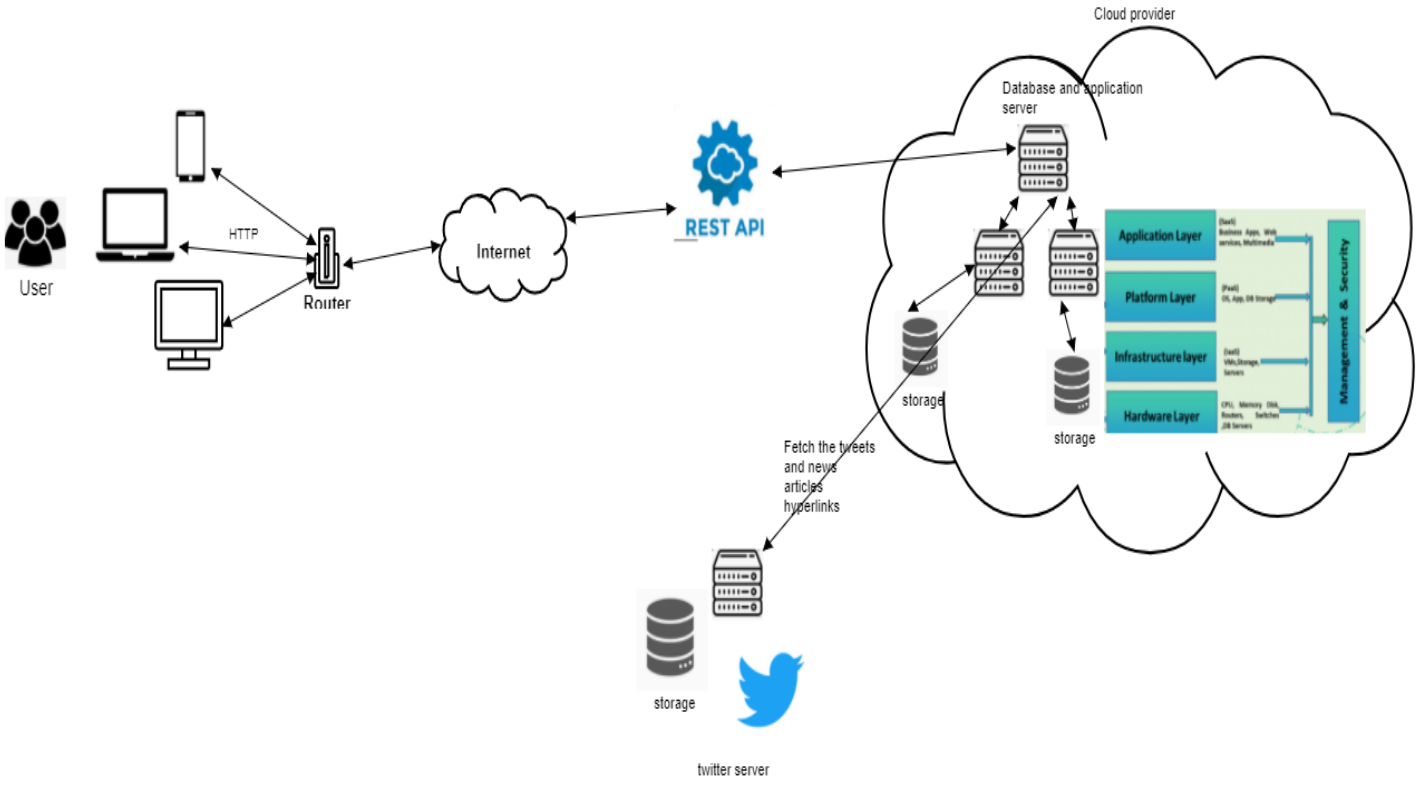


Figure 1: System Architecture for the whole platform

Initially users can visit the platform which will be created by the business clients which are using our API's. The transfer protocol will be HTTP(s) and once the request is sent by the user, the REST API's will transfer the query to the cloud database where the first step is to fetch the tweets from third party vendor i.e twitter, once the tweets are fetched, based on the user requirements if the request is for article then web scraping will be performed as mentioned in the above chapters and then once the required results are received in the cloud server then that input will be given to the respective trained model which will be stored in

cloud itself and then the output will be sent in the form of json which will have the output of the request.

6.2 SEQUENCE DIAGRAM/EVENT DIAGRAMS

The events are divided into 3 categories.

6.2.1 HOME PAGE SEQUENCE DIAGRAM

Initially the users visit the platform and based on the user need he/she will be selecting from the provided options which are specifically the news article and the food reviews these are the business models which are provided by us.

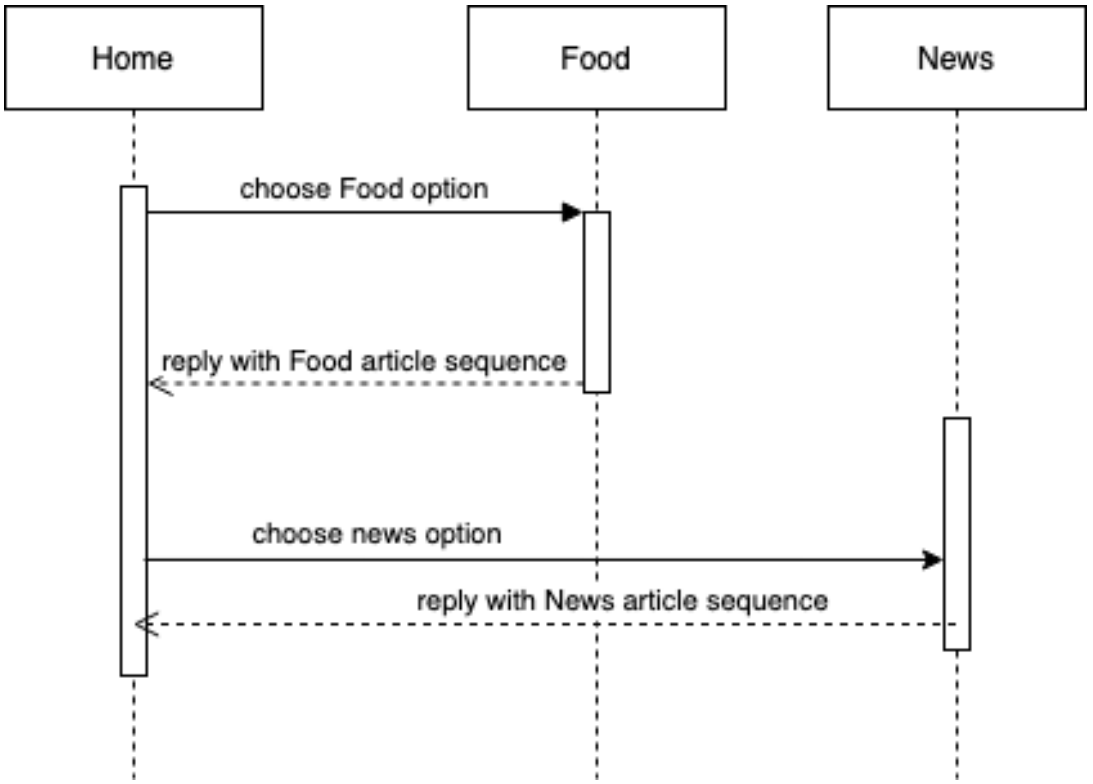


Fig 3: Home Page sequence diagram

6.2.2 NEWS ARTICLE SEQUENCE DIAGRAM:

Once the user selects the required business model and if the business model chosen is news article summarization and emotion detection, the user will lead to the different page where there will be two options either he/she can choose the option of news articles summarization or news articles emotion/sentiment. If the option chosen is summarization , the user will be provided with the selected news channels like CNN, TOI etc and based on their requirements they can select. After this step if the user is interested to select the same summarized data to be given to the emotion detection he can do that otherwise there is a separate option to enter different article/news headlines for emotion detection and the output will be presented on the screen.

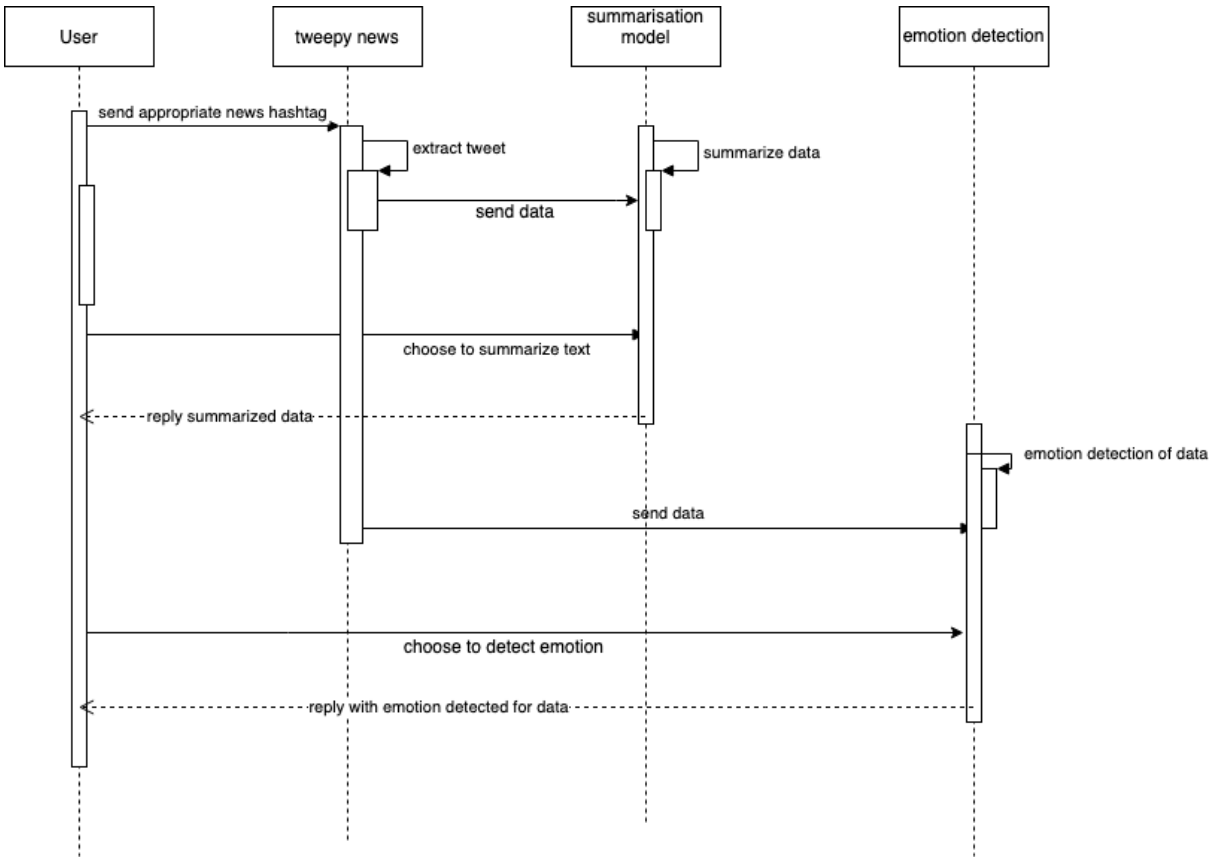


Fig 4: News article sequence diagram

6.2.3 FOOD REVIEWS SEQUENCE DIAGRAM

As mentioned above , once the user selects the required business model and based on the selection he will be redirected to a different page. Once the selected option is Food reviews there will be a text box provided to the user where he/she can enters the hashtags(only hashtags are allowed) and based on the hashtags the tweets will be fetched and based on that the user can decide whether to go for summarization or directly towards emotion detection. After that the respective results will be presented on the screen.

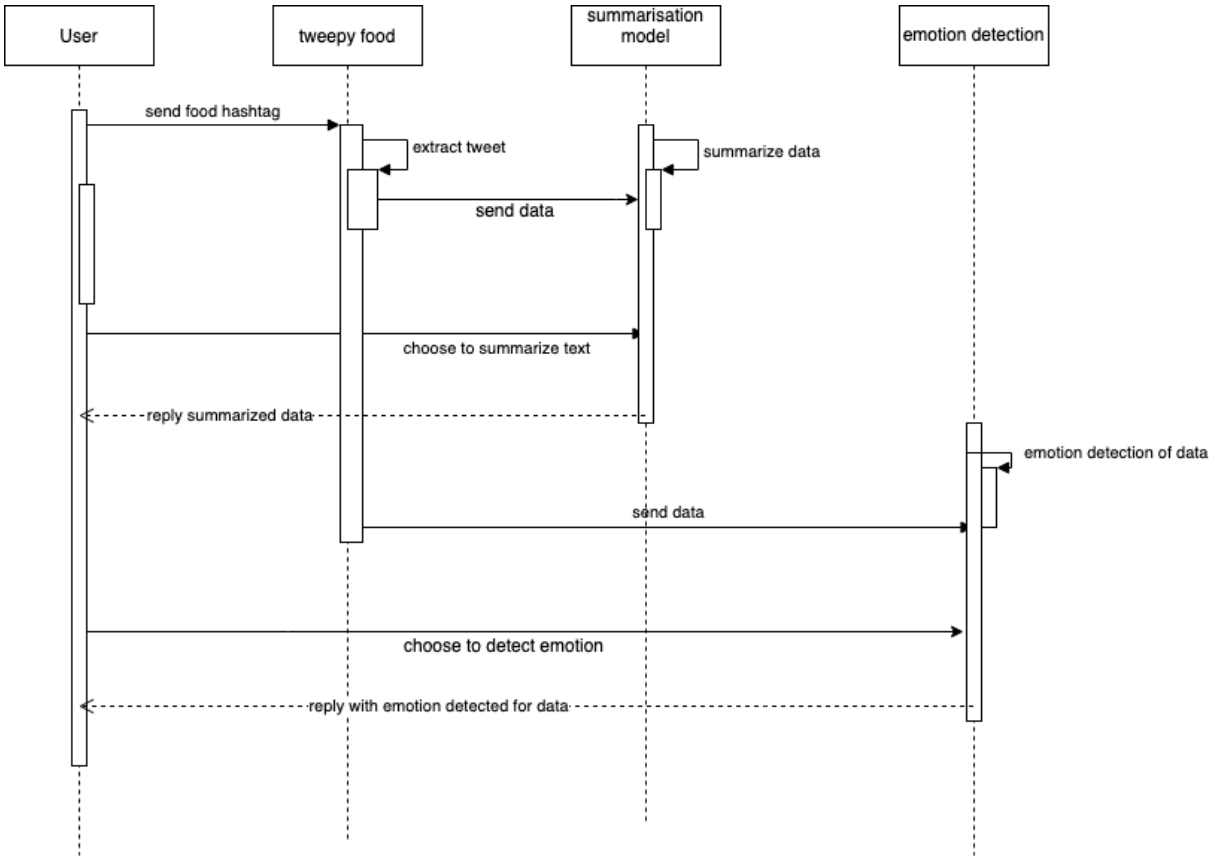


Fig 5: Food Reviews sequence diagram

6.3 DEPLOYMENT DIAGRAM

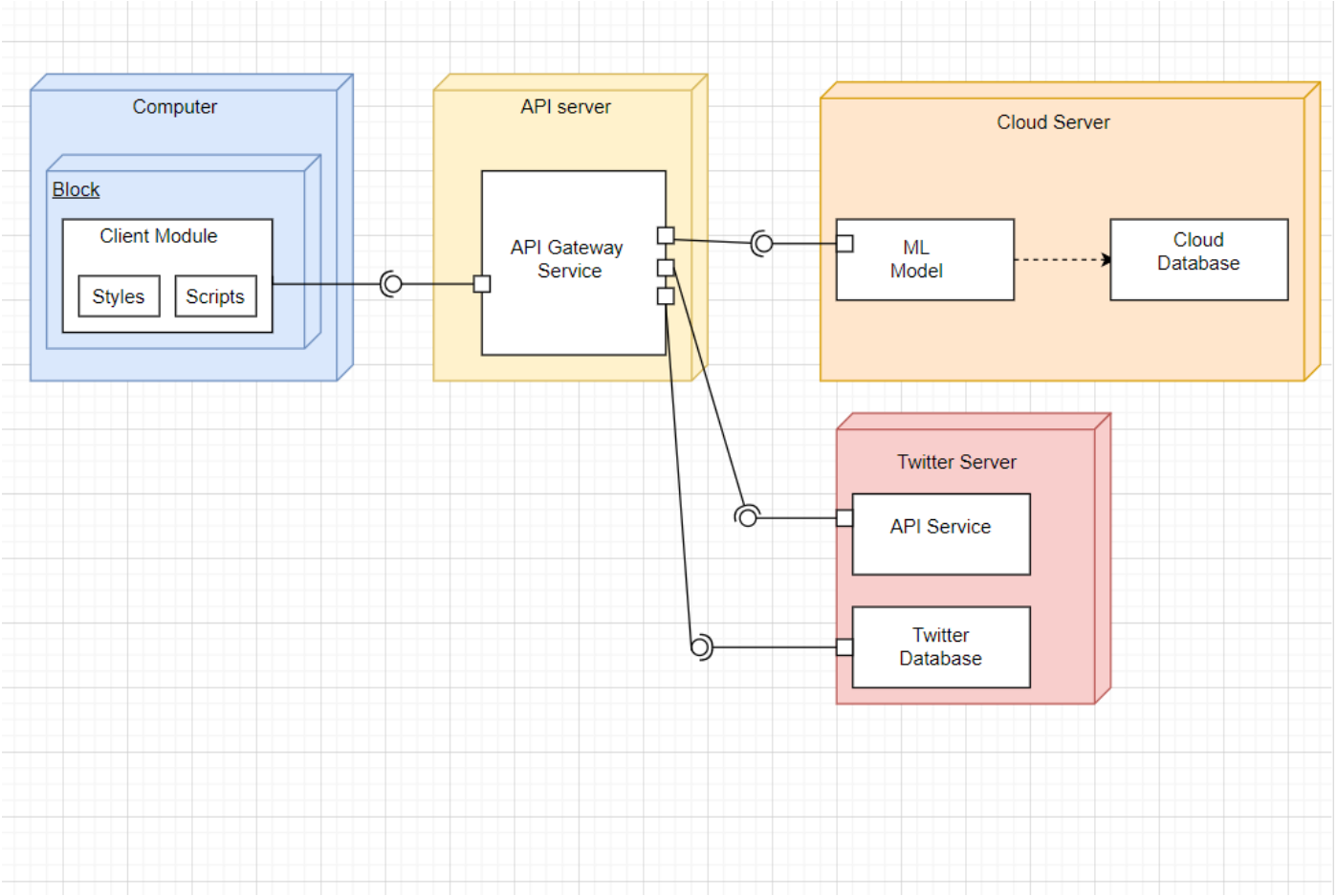


Fig 6: Deployment diagram

This section talks about how the platform will be deployed. The protagonist of the deployment is cloud computing technology. We are using cloud resources extensively ranging from training models, storing datasets and even deployment. In General there are 4 nodes/systems, which are the computer where the user will be requesting, the API server for all the REST API's which will be stored in cloud server, the main cloud server node which will contain all the datasets and required ML models and the last one is the third party node which is the twitter database which we will be using for fetching the tweets

CHAPTER 7

IMPLEMENTATION AND PSEUDO CODE

7.1 Twitter Data Extraction

7.1.1 Configuring Tweepy

We first connect our twitter account to the developer account. We make a note of the "Consumer Key", "Consumer Secret Key", and the access tokens - "Access Token", "Access Token Secret".

We then set up tweepy to authenticate with twitter with the given authentications above.

```
auth = tweepy.OAuthHandler(consumerKey, consumerSecret)

auth.set_access_token(accessToken, accessTokenSecret
```

7.1.2 Initiating the twitter stream

Since we will be streaming data live, we will require a listener class. We will subscribe to the 'TwitterStreamListener' class.

```
listener = TwitterStreamListener()
```

We will now create an instance of the Stream class of tweepy, which will be used to stream all the tweets we need. To this stream class we will pass our authentication in order to connect to our twitter account. We will also pass in our listener instance to this. We then start streaming by invoking the 'filter' method. This will start streaming the tweets based on the terms that we provide and also based on the language that we provide, in this case it is english. The 'filter' method will use the 'filter.json' API and pass any data received to the listener callback.

```
twitter_stream = tweepy.streaming.Stream(auth, listener)

twitter_stream.filter(track=settings.TRACK_TERMS,
languages=["en"])
```

7.1.3 Setting up the listener class

We will now implement our listener class. Firstly, we will derive it from the

tweepy.StreamListener Class.

```
class TwitterStreamListener(tweepy.streaming.StreamListener):
```

We do this because the StreamListener Class has a method called ‘on_data’ which automatically figures out what kind of data Twitter sent and then calls an appropriate method to deal with this. For our tasks we need to override the onStatus() method. This is shown in the code snippet below:-

```
def on_status(self, status):
     if status.retweeted:
          return
     description = status.user.description
     loc = status.user.location
     text = status.text
     # now remove emojis
     text = emoji.get_emoji_regexp().sub(r'', text)
     table = db[settings.TABLE_NAME]
     table.insert(dict(
     user_description=description,
     user_location=loc,
     coordinates=coords,
     text=text,
     user_name=name,
     retweet_count=retweets,))
```

Since retweets can show up the same text many times, hence using computational resources extensively, we will ignore them. All the other lines of code demonstrate the information that we can extract from the user. This information is being stored in a database only for demonstration purposes. Our product will require only the text to be extracted, for this we are doing the same but by removing the emojis in this scenario, as is required for the summarization model.

```
def on_error(self, status_code):
     print('Got an error with status code: ' +
str(status_code))
     return True
def on_timeout(self):
     print('Timeout...')
     return True
```

In the above code snippets, we are overriding the 'on_error' and 'on_timeout' methods to give us an appropriate message when the events given above occur. This makes debugging a lot easier.

7.1.4. Replacing emojis with text

Initially, the demoji has to be installed so as to use it. This can be done by running the below command.

```
pip install demoji
```

We need to initially download emoji related data from Unicode Conostrium's emoji code repository. This is because the emoji list itself will be periodically updated and we need to make sure that we have cached the most recent data so as to process all emojis. The below code snippet shows how to download this data.

```
import demoji

demoji.download_codes()
```

demoji.findall() function takes in the text to be processed as an input, it detects the emojis and emoticons that are present in the input text and gives the description of all of those emojis/emoticons as output.

```
text_input= "'sometimes I'm 😌, and sometimes I'm 😕. But most
of the times I'm 😩 and 🥱'''
demoji.findall(text_input)
```

The output of the above code will be :

```
'😌': 'relieved face'
'😕': 'confused face'
'😩': 'tired face'
'🥱': 'yawning face'
```

As most of the tweets are likely to have emojis, we need to convert these emojis into a text format so that the machine learning models can read these tweets to do their jobs i.e, emotion detection and sentiment analysis. Now that we have the demoji module that does the required task, we can simply replace the emojis with the respective descriptions and pass the tweets on to the machine learning models to carry out the required tasks.

7.2 Abstractive Text Summarisation(Fine-Tuning T5)

We use the concept of transfer learning ,specifically we use pre-trained transformers.As transformers are the current SOA models for various NLP tasks.We specifically use text-text transformer T5.T5 is based on seq-seq model(encoder decoder architecture) and solves various NLP tasks by converting into text-text format.

7.2.1 Defining Pre-Trained Model and Tokenizer

We use 't5-small' for our use case and import the model.Every transformer requires a tokenizer.A tokenizer is nothing but it processes the input text and converts into a sequence of tokens.We also load the specific tokenizer for our transformer.And we load the pre-trained weights by giving the argument as 'summarisation'

```
tokenizer = T5Tokenizer.from_pretrained('t5-small')
model =
TFT5ForConditionalGeneration.from_pretrained('t5-small')

task_params = model.config.task_params
if task_params is not None:
    model.config.update(task_params.get("summarization", {}))

pad_token= tokenizer.pad_token_id
```

7.2.1.1 Model Summary

```
Model: "tf_t5for_conditional_generation_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
shared (TFSharedEmbeddings)  multiple                  16449536
_________________________________________________________________
encoder (TFT5MainLayer)      multiple                  18881280
_________________________________________________________________
decoder (TFT5MainLayer)      multiple                  25175808
=================================================================
Total params: 60,506,624
Trainable params: 60,506,624
Non-trainable params: 0
```

7.2.2 Defining Model Parameters

We define metrics for accuracy optimiser.We use Adam Optimizer as an optimiser for our loss function and learning rate of 1e-08.We use sparsecategoricalcrossentropy as our loss function.

```
optimizer =
tf.keras.optimizers.Adam(learning_rate=learning_rate,epsilon=
1e-08, clipnorm=1.0)
loss_object =
tf.keras.losses.SparseCategoricalCrossentropy(from_logits=Tru
e)
train_loss = tf.keras.metrics.Mean(name='train_loss')

train_accuracy =
tf.keras.metrics.SparseCategoricalAccuracy(name='train_accura
cy')

val_loss = tf.keras.metrics.Mean(name='val_loss')
val_accuracy =
tf.keras.metrics.SparseCategoricalAccuracy(name='val_accuracy
')
```

7.2.3 Defining Train Step

Every tokenizer after taking the input sequence outputs an certain parameters,the important parameters among them is input_ids and attention mask.Input ids are nothing but the words which are represented as set of unique ids.Attention mask is an optional argument,it tells which parts of the input sequence it should give attention to.A set of 1's indicate it should attend to those parts of the input sequence.So the input_ids and attention mask we get from tokenizer are fed into the model during training step .This is nothing but pre-processing step.

```
@tf.function
def train(input,mask, y_pred):
    yids = y_pred[:, :-1]
    labels = tf.identity(y_pred[:, 1:])
    labels = tf.where(tf.equal(y_pred[:, 1:],pad_token), -100,labels)

    with tf.GradientTape() as tape:
        predictions= model(input, attention_mask=mask,
decoder_input_ids=yids,labels=labels, training=True)
```

```
        loss = loss_object(y[:, 1:],predictions.logits)
    gradients = tape.gradient(loss, model.trainable_variables)
    optimizer.apply_gradients(zip(gradients, model.trainable_variables))

    train_loss(loss)
    train_accuracy(y[:, 1:], predictions.logits)
```

We define validation function similar to train ,but we calculate loss based on test data.

7.2.4 Training and Validation

Hyperparamters

```
BATCH_SIZE = 16

SHUFFEL_SIZE = 1024

learning_rate = 3e-5
```

The model was trained for 1 epoch ,with a batch size of 16.We temporarily trained the model on Google Colab for 7hrs.Later the model will be trained on AWS.

The model was evaluated using ROUGE score.ROUGE-1 ,ROUGE-2 and ROUGELSum were calculated on the test data.The achieved results are close to the State of Art.

Metric	Value
ROUGE 1	40.118984089679145
ROUGE 2	18.401294761576324
ROUGE LSum	37.11376539035381

Finally saving the fine-tuned model

```
model.save_pretrained("summarisation_model")
tokenizer.save_pretrained("summarisation_tokeniser")
```

7.2.5 Predicting and Model Demo

The pre-trained model is loaded .Before we summarise we have to clean the text .That is taken care by a function which cleans the given input sentence.Once the input is cleaned we

give it as input to the predict model .THe model uses tokeniser ,gets the input_ids ,attention mask and predicts and gives output_ids.We decode the output ids_s to the predicted sentence.

```
art=''
ids =tokenizer.encode_plus((model.config.prefix + art), return_tensors="tf",
max_length=512)
sum = model.generate(input_ids=ids['input_ids'],
attention_mask=ids['attention_mask'])
pred = [tokenizer.decode(g, skip_special_tokens=True,
clean_up_tokenization_spaces=False) for g in sum]
```

The predicted output is **Englishman anchored jamaica to victory in the men is 4x100m relay gold in moscow . u.s finished second in 37.56 seconds with canada taking bronze . bolt has now collected eight gold medals at world championships .**

We can see that the model has preserved context and importantly the sentence is grammatically correct.

CHAPTER 8

CONCLUSION OF CAPSTONE PROJECT PHASE-1

Social media platforms have emerged one the most influential platforms where people can recommend and share their ideas regarding products, articles, news sports, movies etc hence it becomes important to take the benefit from this and try to implement new features towards business analytics.

Twitter is used by 187 million active users who share their thoughts by posting tweets which shows their interests towards some products and important issues. Those tweets also contain food reviews for certain food items, about their interest and how it tastes and many more. Particular food item is quite famous in some places and some food items in others. If a food eatery wants to start a new business he has to keep all these points in mind.

Sometimes news headlines and articles are too long that people don't try to even read the whole article, hence everyone wants precise and summarized information. Also people don't want their day to start off with some negativity, most of the time people prefer articles which have positivity in it. A news channel has to keep these important points with them while publishing something.

In this project we have tried to cover the above mentioned points and tried to come up with a platform which will ease all the tasks of business analytics and will help the clients to grow their business. We will be creating APIs which can be used by other business clients to implement in their product such as mobile apps, websites etc.

The product will be divided into several stages. The first stage is to take user input regarding what they want to search after that the information will be fetched from twitter based on username or hashtags based on the selected business models. The fetching of tweets is done using tweepy, it is a python library which helps in interacting with all the apis which are required to fetch the tweets and other related twitter information.

The following stage is the summarization and emotion detection model which requires machine learning models and analysis to perform. This required thorough reading of research papers and based on that best and accurate ML model has been chosen. The dataset required for training purposes is one of the best present currently. The datasets can be found online.

All the above steps are done on a cloud platform which makes this project more unique as this whole project is cloud assisted and finally the performance evaluation will be measured which will help us in doing further studies in the cloud computing field and how to deploy such projects. This project helped us to identify multiple interesting fields and how to aggregate those into one platform which will help the users and make the process easier.

CHAPTER 9

PLAN OF WORK FOR CAPSTONE PROJECT PHASE-2

After six months of thorough reading and finding the interesting topics and starting implementing things it's now the final stage of the project where we have to implement a major part of the project. Capstone phase 1 helped us in building basic blocks of the project, we created system design and developed the ML models. Now it's the stage where we have to implement everything in a single module.

Capstone phase 2 will start with documenting the deliverables which we have to submit. All the tasks will be done parallely. Fetching the tweets and then extracting the meaningful sentences will be done and then passing it to the summarization model which is completed in capstone phase 1 will be delivered.

Once the summarization is done successfully then building the emotion detection/ sentiment analysis model has to be done in this phase. This model will be built from scratch in this phase itself. Once this model is done we will be testing with the summarized output of the previous model and analyse the output and will try to change the parameters accordingly.

The final stage of the project will be to make APIs and test them in POSTMAN software and generate proper documentation for those APIs. A website in the form of demonstration purpose will be made to show the use cases of the APIs. All these modules will be migrated to the cloud platform once its done locally.

Once the project will be cloud hosted , performance evaluation will be done and all the necessary insights will be analysed. A performance evaluation report will be presented in terms of usage, time , power consumed etc, which will mark the end of the project.