Telecom Sentinel: Twitter Sentiment Analysis of Customer Experience Feedback of Singapore Telecom Companies

Akshay Sachdeva[1], Aniket Mohan Arasanipalai[2], Siddhant Naveria[3], and Tarun Rajkumar[4]

[1,2,3,4]NUS-Institute of System Science

***Abstract*— Corporations have always required timely customer experience reviews on their products to change current pricing and policies in order to keep ahead of their rivals. You will create a positive customer experience by assessing and acting promptly on consumer feelings. Social networks such as Twitter represent general public opinion and collective intelligence and can therefore be harnessed. They have evolved as a resource for extracting sentiments for applications in various fields. Sentiment analysis can be used to obtain the overall customer experience of a large customer base on a real time, as well as identifying key aspects of negative or positive sentiments. In this project, we introduce a BERT machine learning model to classify sentiments of different tweets extracted. The results between different machine learning models are also compared. The different preprocessing steps are also described which are done to ensure maximum accuracy and efficacy. The sentiment analysis results can be used by managements to take timely actions for improving the future customer experience and avoiding customer churn.Our proposed system shows true effectiveness as it presents the results obtained on a user friendly dashboard allowing the business stakeholders to view results without any hassle. With this, we have also integrated an intelligent chatbot that that accommodates for differences in syntax of questioning grammar but is still able to return a relevant response based on keywords in the query via Natural Language Processing (NLP).**

Keywords: Social Network | Text Mining | Customer Experience | Sentiment Analysis | Deep Learning | Flask | Dashboard

I. INTRODUCTION AND MOTIVATION

As a microblogging and social networking website, Twitter has become very popular and has grown rapidly. An increasing number of people are willing to post their opinions on Twitter, which is now considered a valuable online source for opinions. As a result, sentiment analysis on Twitter is a rapid and effective way of gauging public opinion for business marketing or social studies. For example, a business can retrieve timely feedback on a new product in the market by evaluating people's opinion on Twitter. As people often talk about various entities (e.g., products, organizations, people, etc.) in a tweet, we perform sentiment analysis at the entity level; that is, we mine people's opinions on specific entities in each tweet rather than the opinion about each whole sentence or whole tweet. We assume that the entities are provided by the user, e.g., he/she is interested in opinions on iPhone (an entity).

Among social network websites, Twitter is one of the most popular. We select Twitter for crawling user responses as it has many registered users and an enormous amount of their content and also that Twitter provides restricted access to tweets to the general public via APIs. Twitter users generate tweets of the order of 500 million per day.

With the explosive growth of social media (e.g., reviews, forum discussions,blogs, micro-blogs, Twitter, comments, and postings in social network sites)on the Web, individuals and organizations are increasingly using the content in these media for decision making. Nowadays, if one wants to buy a consumer product, one is no longer limited to asking one's friends and family for opinions because there are many user reviews and discussions in public forums on the Web about the product. For an organization, it may no longer be necessary to conduct surveys, opinion polls, and focus groups in order to gather public opinions because there is an abundance of such information publicly available. In recent years, we have witnessed that opinionated postings in social media have helped reshape businesses, and sway public sentiments and emotions, which have profoundly impacted on our social and political systems.

Corporations spend a huge amount of money and time on brand monitoring and collection of real-time customer experience feedback to keep a watch on their sales and revenues. Most of the traditional methods of monitoring and feedback have a huge cost involved with them and have latency issues. The feedback results are summarized and presented to competent authorities of higher management after a delay of a few days to a couple of weeks from the actual start of the feedback gaining process. Within this span, market scenarios and equations get changed, therefore requiring a process to obtain feedback in quick turnaround time.

There has been a large amount of research in the area of sentiment classification. Traditionally most of it has focused on classifying larger pieces of text, like reviews. Tweets (and microblogs in general) are different from reviews primarily because of their purpose: while reviews represent summarized thoughts of authors, tweets are more casual and limited to 140 characters of text. Generally, tweets are not as thoughtfully composed as reviews. Yet, they still offer companies an additional avenue to gather feedback. There has been some work by researchers in the area of phrase level and sentence level sentiment classification recently.

Manual classification of millions of posts for opinion mining tasks is an unfeasible effort at human scale. Several methods have been proposed to automatically infer human opinions from natural language texts. Due to the inherent

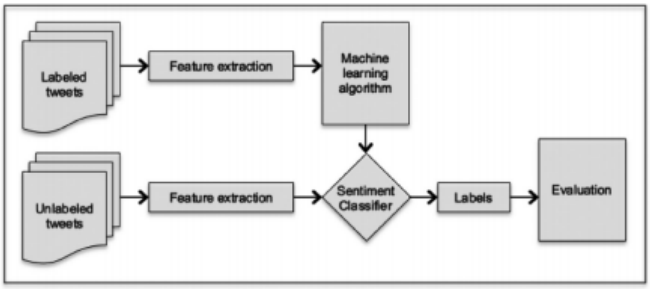

Fig. 1. Typical process for Sentiment Classification

subjectivity of the data, this problem is still an open problem in the field.

Another challenge of microblogging is the incredible breadth of topic that is covered. It is not an exaggeration to say that people tweet about anything and everything. Therefore, to be able to build systems to mine Twitter sentiment about any given topic, we need a method for quickly identifying data that can be used for training.

Many novel SA methods have been specially developed for TSA. These methods can be mainly divided into two categories: fully supervised methods and distantly supervised methods. The fully supervised models try to learn the classifers from manually labelled data. Recent work have focussed on distantly supervised methods which learn the classifiers from the data with noisy labels such as emoticons and hastags.

In this project we explore tweets of consumers who prattle and voice their experience with telecom products and services. Our exploratory proposal in addition to sentiment analysis, is to further granularize the specific superlative and abysmal services/products associated with these tweets. We introduce sentiment classification through the BERT transfer learning mechanism as well as through transformer and compare accuracy between the two to understand, which can be used in a more effective way. As a product, to enhance business intelligence we are developing a dashboard which will provide a visual representation of this enormous data. We employ an entity-level sentiment analysis approach to the Twitter data.

Further to ease the interaction of business users, we will provide a conversational user interface. We believe that our method is more desirable for practical applications due to its nature of no manual involvement and its ability to automatically adapt to new fashions in language, neologisms and trends.

The remainder of this document is structured as follows. Related studies and background are discussed in Section II. Objectives and Success Measurements is presented in Section III. The solution scope is presented in Section IV. Implementation Discussion is covered in Section V. Finally, in Section VI, we conclude the project.

II. RELATED WORK

A. Social Networks

The diffusion of information on social media is at an enormous scale and real-time analysis techniques focus on it to provide predictive inferences [1]. People prefer the social network endorsement of brand over other types of endorsements [2], [3]. Social networks provide electronic word of mouth option to the public, which influences them more and its measurement is also feasible [4], [5]. Researchers discuss how people seek opinions from others be it everyday chores or national elections emphasizing the need for opinion mining. Respondents may not be comfortable with the traditional feedback methods or in some cases don't have time for feedback surveys. Respondents may not report their true opinions while filling written questionnaires, but they let out their feelings on social networks when they like or dislike certain brands and companies. In the research carried out, social network sentiment analysis has been used to provide a solution to this problem. Social network data can be fetched real-time and requires less time and effort to summarize and lead to conclusions.

Twitter uses Open Authentication (OAuth) for APIs requesting information from it [6]. The crawled data includes pairs of userids, tweet, the relation between userids, relationship date, hashtags in Tweet etc.

B. Sentiment Analysis

The proposed research is in the area of sentiment analysis. To determine whether a document or a sentence expresses a positive or negative sentiment, two main approaches are commonly used: the lexicon-based approach and the machine learning-based approach.

The lexicon-based approach [7] determines the sentiment or polarity of opinion via some function of opinion words in the document or the sentence. As discussed earlier, this method can result in low recall for our entity-level sentiment analysis. The machine learning-based approach typically trains sentiment classifiers using features such as unigrams or bigrams [8]. These methods need manual labeling of training examples for each application domain.

There are also some approaches that utilizes both the opinion words/lexicon and the learning approach. For example,[9] used a subjectivity lexicon to identify training data for supervised learning for subjectivity classification. Our work does not do subjectivity classification. A similar idea was also applied to sentiment classification of reviews in [10], which classifies reviews into two classes, positive and negative, but does not have a neutral class. Supervised learning is the dominant approach. [11] built a sentiment classifier to classify tweets into positive, negative and neutral classes.

By itself, sentiment analysis systems are often characterized into statistics-based and knowledge-based systems. On the one hand, statistical approaches have proven to be generally semantically feeble. This is due to the fact that

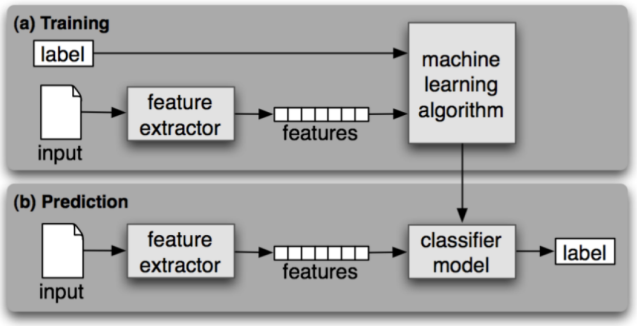

Fig. 2. Generalized Sentiment Classification through Machine Learning

statistical text classifiers only work with adequate precision when given a satisfactorily vast text input. Although statistical approaches are able to affectively classify users' text on the page or section level, they do not work properly on smaller text parts such as sentences.

On the other hand, concept-level sentiment analysis is a task which relies on large semantic knowledge bases which has recently growing interest within the scientific community as well as the business world. It emphasizes on a semantic text analysis through the use of web ontologies or semantic networks which enable an aggregation of the conceptual and affective information associated with natural language opinions.

The analysis at concept level enables to infer the semantic and affective information associated with natural language opinions and, well ahead, to enable a comparative Fine-grained feature-based sentiment analysis.

III. Objectives and Success Measurements

A. Objectives

The objective of this project is to develop a sentiment analysis model by extracting data from twitter. The model should be able to accurately classify sentiments into positive and negative.

B. Success Measurements

It is vital to understand the success of the model based on a few measures. There are a few key measures of our model:

- Whether the sentiment model was able to provide a good accuracy
- Whether the sentiment model was not error prone
- Whether the model could be tweaked easily for improvement
- Whether the dashboard and chatbot was responsive

IV. Solution

A. Target Audience

The major target audience would be business stakeholders who would be interacting with the dashboard and chatbot on a day to day basis to visualize the numbers, tweets and request for insights.

B. Data Quality and Knowledge Base

Data is scraped from Twitter with respect to different telecommunication networks of Singapore like Singtel, Starhub and M1.

C. Project Scope

In this project, a sentiment analysis based system is developed based on extraction of Twitter data of different users of telecommunication networks like Singtel, Starhub and M1. Sentiments of the tweets are classified as 'Positive' and 'Negative', and the information related to tweets are presented on a dashboard. The user can interact with the dashboard through a chatbot. The intent-based system can answer questions with context via rich messages response as well as voice format

<insert mind map here>

The domain diagram above arranges the factors affecting the user's question flow in a structural format. This chatbot can be broken down into multiple layers of different components before arriving at the answers. These question flows are gathered from users having a proposed question flow and represent their inherent preference.

V. System Architecture

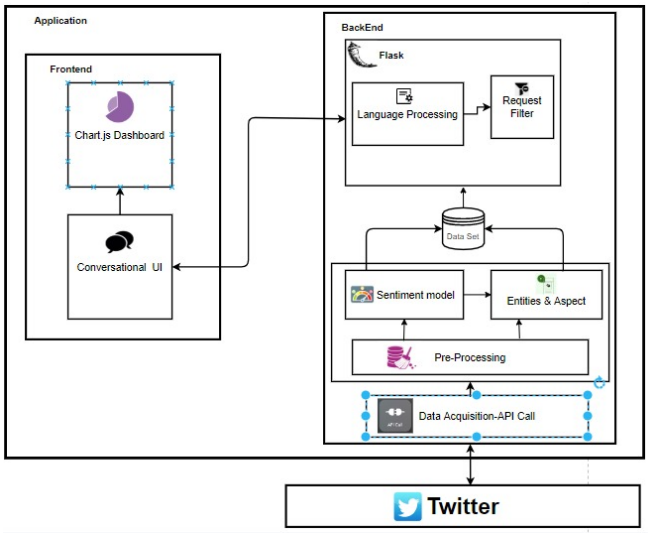

Fig. 3. System Architecture

The Application is divided into 2 parts - front end and back end. There are mainly 2 components in front end: A Conversational UI (chat screen) and Dashboard to showcase the results asked over the chat. The backend has 3 main components:

- Flask module, which handles the request from front end and send the response back to the front end. After receiving the request, the language processing component fetches the intent of the conversation and

passes the intent to the request filter component, to query required information from the dataset
- The sentiment modules extract pre-trained models, which we have trained for sentiment analysis and entity and aspect extraction. There is a pre-processing unit which cleans the data (stemming, lemmatization, stop word removal, tokenization etc.) and transforms the raw data into the required format. The output of models are saved in data set to be used for future queries.
- The API call module comprises of a module which makes the API call to twitter to fetch the tweets related to the telecom industries. This module acts as the input to the sentiment module unit.

The dashboard is designed keeping in mind the KPIs, which include:

- Tweets timeline that references
- Distribution of tweets based on sentiments
- Grievances associated with negative tweets

VI. Implementation

A. Dataset Creation

In this project, tweets for - singtel, Singtel Singapore - the Twitter Handles for the telecom company were extracted on a daily basis from 1st January 2019 to 31st December 2019. This period is announced as other competitors launched offers to counter Singtel's presence in Singapore. This could be a real test for Singtel to analyze if the general public is inclined towards its services or not amidst the presence of operators like Starhub, TPG, and M1.

The combined sentiment of the general public can be obtained to check if the general public is showing interest towards the new entrant TPG or existing market leaders, when all of them were now offering mobile services at similar competitive tariffs.

For annotatating the data, we have used Google Cloud Platform's(GCP) NLP service. Due to this, the accuracy of our model can reach a maximum accuracy that is representative of the accuracy provided by the GCP service.

B. Data Preprocessing

Firstly, in pre-processing we remove the blank rows from the dataset. The dataset scraped with twitter data has numerous special characters, which are considered as junk for our training mechanism. We have removed these special characters and url present in the text. The next step in pre-processing is to perform tokenization and lemmatization. For tokenization, we have used BERT tokenizer to tokenize text in the format supported by the BERT model. Post this, We have used Lemmatization to extract lemmas for the grievance extraction.

All the non-word tokens are omitted and only the text is considered for this task.

C. Sentiment Analysis using Fine-Tuned Transformer

Transformer is an architecture for transforming one sequence into another one with the help of two parts (Encoder and Decoder),which utilizes self-attention to compute representations of its input and output without using sequence-aligned RNNs. In this way, it reduces the number of operations required to relate signals from two arbitrary positions to a constant number and achieves significantly more parallelization.

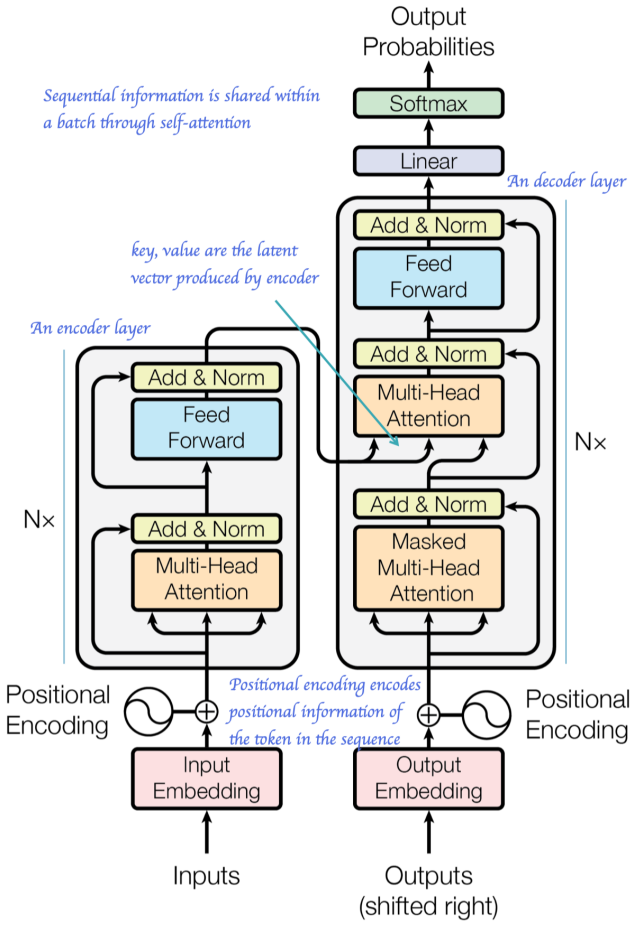

Fig. 4. The Transformer Model Architecture

Fig. 4. explains the model architecture for Transformer, originally from paper Attention is All You Need.The Transformer follows the encoder-decoder structure using stacked self-attention and fully connected layers for both the encoder and decoder, shown in the left and right halves of the following figure, respectively. Positional Encoding: Sine and cosine functions of different frequencies are used to encode the position information:

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

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2 to 100002.

Encoder and Decoder Stacks: The encoder is composed of a stack of N=6 identical layers. Each layer has two sublayers. The first is a multi-head self-attention mechanism(we will come back to it soon), and the second is a simple fully connected feed-forward network. Residual connections are employed around each of the two sub-layers, and layer normalization is applied in between. That is, the output of each sub-layer is x+Sublayer(LayerNorm(x)). The decoder is also composed of a stack of N=6 identical layers. In addition to the two sub-layers in the encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack (i.e., where we have the output of the encoder as keys and values). Sub-layers in the decoder follows the same fashion as that in the encoder.

Attention: Q is a matrix that contains the query (vector representation of one word in the sequence), K are all the keys (vector representations of all the words in the sequence) and V are the values, which are again the vector representations of all the words in the sequence. For the encoder and decoder, multi-head attention modules, V consists of the same word sequence than Q.

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

The intuition behind pre-trained language models is to create a black box which understands the language and can then be asked to do any specific task in that language. The idea is to create the machine equivalent of a 'well-read' human being.

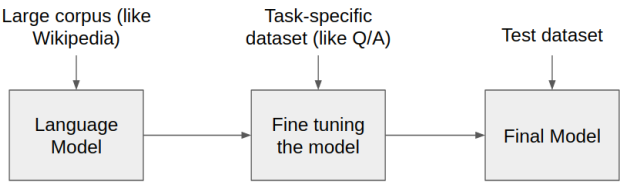

Fig. 5. Pretrained Model Process.

The language model is first fed a large amount of unannotated data (for example, the complete Wikipedia dump). This lets the model learn the usage of various words and how the language is written in general. The model is now transferred to an NLP task where it is fed another smaller task-specific dataset, which is used to Fine Tuned and create the final model capable of performing the aforementioned task.

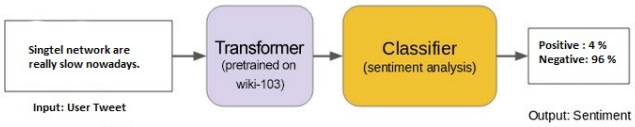

Fig. 6. Transformer Fine-Tuning on a downstream task.

Text Processing: To encode the texts to sequences of integers so they can be processed by our neural network model, the tokenizer used for processing the data at the pretraining stage is used.i.e. pytorch-transformers's BertTokenize. pre-processor is responsible to encode text as integers, followed by padding or truncating so they are all the same length. This number is determined by the Transformer as Fine-Tuned of the model is required and is a hard limit in this case. A special classification token[CLS] is appended to the end of each sequence. This token's hidden state will be fed to our classification head at Fine-tuning. Labels are converted to integers using the label2int mapping.

Model: The architecture and pretrained weights are used from NAACL 2019's Transfer Learning model. The base model is an OpenAI GPT-style Transformer, which is extended with a single linear layer with num-classes (here 2) output neurons, forming Transformer With ClfHead

Fine-Tuning and Evaluation: For compact yet full-featured training and evaluating pytorch-ignite is extremely convenient. Simply,trainer and evaluator Engines let us to call our update functions on each batch of an iterator.Accuracy as our main metric is attached and evaluated by the validation set after each epoch. A custom learning rate scheduler, a progress bar and model checkpointing is also added to the trainer.

D. Sentiment Analysis using BERT

BERT (Bi-directional Encoder Representations from Transformers) was proposed by Google last year and was able to single-handedly achieve SOTA performances on 11 separate NLP tasks!! It has since been the source of multiple language models spawning from it.With the help of sentiment analysis systems, the unstructured information could be automatically transformed into structured data of public opinions about products, services, brands, politics, or any topic that people can express opinions about. This data can be very useful for commercial applications like marketing analysis, public relations, product reviews, net promoter scoring, product feedback, and customer service.

Sentiment analysis systems allows companies to make sense of this sea of unstructured text by automating business processes, getting actionable insights, and saving hours of manual data processing, in other words, by making teams more efficient.Aspect level sentiment analysis focus on three important aspects in an opinion; opinion expression, opinion targets and opinion holder.

Sentiment analysis can be applied at different levels of scope:

- Document level sentiment analysis obtains the sentiment of a complete document or paragraph.
- Sentence level sentiment analysis obtains the sentiment of a single sentence.
- Sub-sentence level sentiment analysis obtains the sentiment of sub-expressions within a sentence.

There are different kinds of deep learning methodologies can be used to classify like Neural Networks, LSTM, etc.

To create more effective models in accordance with newer model building methods, we have built models based of Transfer learning. It is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

In our project, we have made use of the BERT(Bidirectional Encoder Representation of from Transformers) algorithm. BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training.

Sentiments are classified into positive and negative(1 or 0) based on their polarity scores.

A high-level illustration of our approach to twitter sentiments classification is shown in the fig. below7 . Firstly, we used twitter data and perform pre-processing to remove the noise from the twitter text. Further, we tokenize the input tweets using Bert tokenizer and then, we utilize BERT to encode the input sequence into a vector that contains semantic and sentiment information. Finally, the output vector is fed into a sentiment classifier for sentence-level sentiment classification.

For an input sequence of tokens w1,w2,...,wN, BERT first constructs the token representations E1,E2,...,EN by summing the token embeddings, segment embeddings, and position embeddings. The token representations are then encoded by a stack of identical layers. Each layer consists of a multi-head self-attention sublayer and a position-wise fully-connected sublayer. Specifically, for a sequence of hidden states H=h1,h2,...,hN, the output of the multi-head self-attention sublayer S=s1,s2,...,sN is calculated as follows:

$$aij^(k) = Softmax(1/\sqrt{ds}(WQ^(k)hi)^T(WK^(k)hj)) \quad (1)$$

$$si(k) = \sum j = 1^N aij^(k)(WV^(k)hj) \quad (2)$$

$$si = WO[si(1), si(2), \ldots, si(K)] \quad (3)$$

After residual connection, and layer normalization, the processed output S is denoted as follows

$$oi = W2ReLU(W1si + b1) + b2 \quad (4)$$

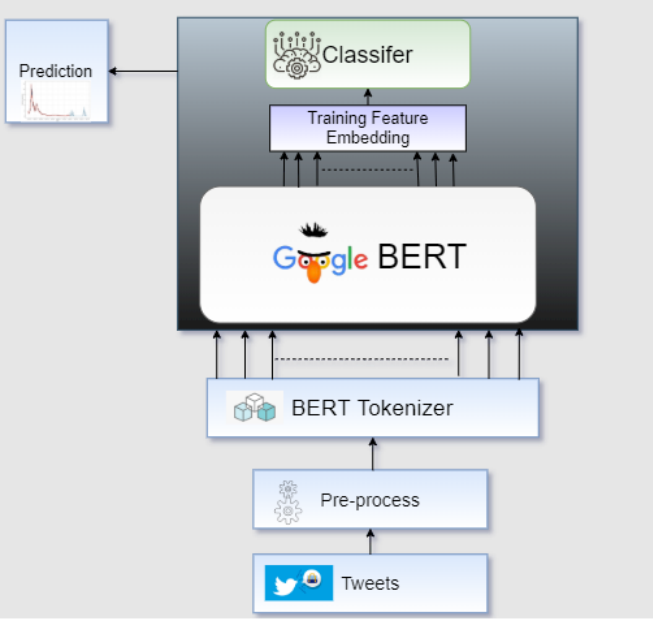

Fig. 7. BERT Sentiment Classification Process.

Similarly, residual connection and layer normalization are applied to the output and the processed output is denoted as follows

$$O = LayerNorm(S + O) \quad (5)$$

Finally, the output of the layer is used as the contextual representation of the input sequence.

For classification of sentiments we have used transfer learning by using pretrained Bert model and applying classifier as a last layer. The last layer is a linear layer with logistic softmax function as activation function for the output. The output from Bert model act as training features embeddings, which is passed to the last layer. The last layer act as a classification layer and. Output from this layer is the logarithmic probability of the classes (positive or negative).

E. Chatbot

In order to ease interaction of the data to business owners and provide inference from the data, we have created a chatbot to interact with the system in addition to viewing the dashboard. We incorporated the chatbot that answers questions explicitly from the data being visualized.

A chatbot is an artificial intelligence software that can

simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, mobile apps or through the telephone, etc. These chatbots can be used in various industries for different purposes. E.g. What is the most negative product? What was the timeline for the most negative tweets?

Data Pre-processing for Intent Detection can be seen in Fig. 5.

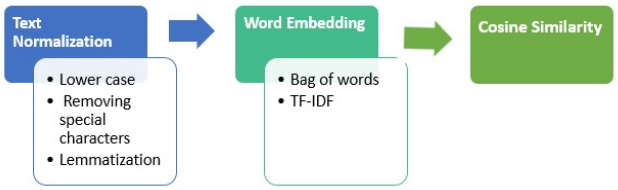

Fig. 8. Chatbot Pre Processing Process

Text normalization converts the data into lower case and removes special characters, Extract relevant content from a Series, Check NaN values and then perform lemmatization, which usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.

Word Embedding: The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model is simple to understand and implement and has seen great success in problems such as language modelling and document classification. It involves two things: A vocabulary of known words: This step revolves around constructing a document corpus which consists of all the unique words in the whole of the text present in the data provided. It is sort of like a dictionary where each index will correspond to one word and each word is a different dimension. 2. A measure of the presence of known words.

tf–idf or TFIDF, short for term frequency-inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

Term frequency: It is used in connection with information retrieval and shows how frequently an expression (term, word) occurs in a document. Term frequency indicates the significance of a particular term within the overall document. It is the number of times a word wi occurs in a review rj with respect to the total number of words in review rj.

$$TF(w_i, r_j) = \frac{\text{No. of times } w_i \text{ occurs in } r_j}{\text{Total no. of words in } r_j}$$

Inverse document frequency: The inverse document frequency is a measure of how much information the word provides, i.e., if it's common or rare across all documents. It is used to calculate the weight of rare words across all documents in the corpus. The words that occur rarely in the corpus have a high IDF score.

$$\operatorname{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

- N: total number of documents in the corpus $N = |D|$
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\operatorname{tf}(t, d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

TF-IDF is calculated as:

$$\operatorname{tfidf}(t, d, D) = \operatorname{tf}(t, d) \cdot \operatorname{idf}(t, D)$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. Since the ratio inside the IDF's log function is always greater than or equal to 1, the value of IDF (and tf–idf) is greater than or equal to 0. As a term appears in more documents, the ratio inside the logarithm approaches 1, bringing the IDF and tf–idf closer to 0. TF-IDF gives larger values for less frequent words in the document corpus. TF-IDF value is high when both IDF and TF values are high i.e the word is rare in the whole document but frequent in a document. TF-IDF also doesn't take the semantic meaning of the words.

Building the Deep Learning Model:

A deep learning model is used to train sentences to enable responses to be provided accurately. The model is built using natural language data into a training set and using a Keras sequential neural network to create a model. Keras is a high-level neural networks API, capable of running on top of Tensorflow, Theano, and CNTK. It enables fast experimentation through a high level, user-friendly, modular and extensible API. Keras can also be run on both CPU and GPU. The easiest way of creating a model in Keras is by using the sequential API, which lets you stack one layer after the other. The problem with the sequential API is that it doesn't allow models to have multiple inputs or outputs, which are needed for some problems. In the first Dense layer which is a hidden layer, 3 arguments are given which are 5000 as the numbers of neuron on that layers, input shape

which is gonna be the size of our bow matrix and relu as activation function. The activation function is the one who Tuned the output of a neurons. So for example the output of our neuron is expected to contain no negative value, but the neuron produce an output of -0.87.By using relu as activation function, it means that function f(x)=max(0,x) which in plain english means that any negative number will be converted into 0, so in the given example the final output of the neuron is 0 not -0.87.

For the second layer, dropout layer is added to prevent overfitting. Theoretically dropout layer could prevent the overfitting by randomly ignore neuron(s) in each iteration meaning that for the ignored neuron, the weights will not be updated and temporally removed from the training process. The intuition behind dropout is that by temporary remove a neuron from training process, it will make the weights adjustment for other neuron changes differently compared to normal condition which the neuron is included. If the model will overfit in the end of the training, it could be prevented because the weights is adjusted in different ways than the overfit weights.

For the last layer which is the output layer, the number of neuron is equal with the number of class we have and softmax activation function is used. After the layers of our model are deFined, in Keras compiling the model is required and defining the loss function, optimizer and metrics. The loss function will measure the performance of our model in training process and measure the learning process. Smaller loss means that our model is closer to converge with training data. Optimizer in machine learning refer to method, rule or formula that govern how our model minimize the loss during training. Because optimizer job is to govern oh how to update the weights in Neural Network,

F. Dashboard

The dashboard and the different graphs of the metrics are created in chartJS. Chart.js is a community maintained open-source library that helps in visualizing data through Javascript. It supports different chart type and the ease of which graphs which can be created, and the responsiveness help in favoring this library.

The base design was developed over the dashboarding template using app seed flask template. As part of customisation for Sentinal we've developed our own architecture seperating the frontend and the backend, the primary reason being in issues with flask threads synchronization with tensorflow models.

Frontend is a flask application designed in MVC pattern. Instead of cramming the code into one place, Flask helps to organize (1) logic, (2) design, and (3) database into separate files.

LOGIC: 'main.py' imports the Flask module, creates a web server, creates an instance of the Flask class. DESIGN: The Flask Framework looks for HTML files in a folder called templates .DATABASE: Flask does not support databases natively, but there are a number of Flask extensions such as Flask-SQLAlchemy.

Each page/tab is a view served by a 'route' from the flask application, and each page/tab designed as a model and the single page application the container. As for the various graphs being used in the application, they use chartjs as the base with our own customisation.Given the separation of the frontend and backend, all the data required for the charts, metrics and tables are fetched by making an API call from the frontend to the backend.

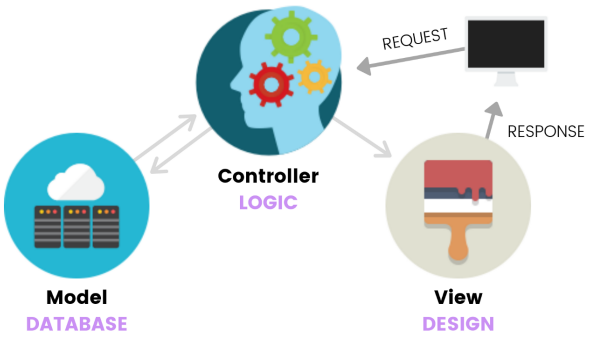

Fig. 9. Flask Structure

The chatbot in the frontend is essentially a html form making api calls to the backend and displaying the response. Other libraries used for the frontend included Bootstrap, which consists of CSS and JavaScript files. It's a framework that makes Website development faster and easier. Jquery, which takes common JavaScript tasks and wraps them into methods. Then, instead of writing out all that code by hand, calls can be made to these methods popper and perfect-scrollbar. Backend is the brains of the operations and acts as an API service. The API calls range from fetching tweets, user metrics, tweets metrics, sentiment scores to processing and predicting chat conversation and answers.

The issues faced pertaining to the version of keras and tensorflow and how they handle threads in a flask app, while it has been managed to ensure that the session and the graphs being used for the model stays persistent and on a single thread this is a bug yet to be fixed by tensorflow.

VII. AWS Deployment

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. The front-end and back-end of the application is dockerized into separate containers and pushed to Amazon EC2, to provide ease of use for the users. Also, Continuous Integration and Deployment pipline is build to support changes instantly over the application on cloud. Travis CI is a Continuous Integration service used to build and test software projects hosted at GitHub. So, when the commit is pushed to the master branch of the project hosted

at GitHub, Travis CI will run tests and build the latest commit of master branch.

VIII. Results

The following tables compare the results, accuracies between all the above mentioned models, and these results would indicate which proved to be the most effective model.

The sentiment analysis model results are as shown in the table below:

TABLE I
EVALATING RESULTS BETWEEN BERT AND FINE-TUNED TRANSFORMER

Comparison between BERT and Fine-Tuned Transformer				
Models	AUC	Accuracy	F1 score	FN
BERT	0.818	0.823	0.844	228
Fine-Tuned Transformer	0.785	0.732	0.767	339

Table 1 above evaluates the results obtained as part of sentiment analysis from the BERT and the Fine-Tuned Transformer models. The BERT model provides AUC and Accuracy of 0.818 and 0.823 while the Fine-Tuned Transformer model provides results 0.785 and 0.732 respectively. It can be seen that the performance of BERT is better than that of Fine-Tuned Transformer due to the results obtained, making this a reason why we chose BERT over Fine-Tuned Transformer. We can also see that the F1 score which is a balance between precision and recall has a higher value for BERT compared to the Fine-Tuned Transformer.

TABLE II
EVALATING RESULTS BETWEEN BERT AND FINE-TUNED TRANSFORMER

Comparison between BERT and Fine-Tuned Transformer				
Models	False Positive	Loss	Precision	Recall
BERT	298	0.531	0.826	0.861
Fine-Tuned Transformer	446	0.169	0.742	0.79

The following results table, table 2 further evaluates the two models based on a few other key metrics like False Positive, Loss, Precision and Recall. With Precision and Recall being higher for BERT, it makes the model more performance centric compared to the Fine-Tuned Transformer.

TABLE III
EVALATING RESULTS BETWEEN BERT AND FINE-TUNED TRANSFORMER

Comparison between BERT and Fine-Tuned Transformer		
Models	True Positive	True Negative
BERT	1031	1423
Fine-Tuned Transformer	906	1289

Table 3 below depicts the true positive and true negative values for the different models. The values being higher for BERT make it statistically significant compared to the Fine-Tuned Transformer.

TABLE IV
EVALUATING RESULTS OF DEEP NEURAL NETWORK BASED CHATBOT

Deep Learning Model for Chatbot				
Accuracy	Loss	MSE	MAE	Cosine Promimity
0.955	0.130	0.002	0.005	0.965

Table 4 consists of different metrics like Accuracy, MAE, Loss, and Cosine similarity etc to evaluate the deep neural network for the chatbot. From the above table, we see that deep neural network works effectively in speech responses. With Accuracy being high, we can observe that the model is able to recognize different speech utterances. With the Mean Square Error(MSE) and Mean Absolute Error(MAE) values being low, the information that can be obtained shows that the data values are dispersed closely to its central moment (mean); which is usually great and something which is desirable.

Cosine similarity is a metric used to determine how similar the documents are irrespective of their size.

Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. When plotted on a multi-dimensional space, where each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents and not the magnitude.

With the value here, being above 0.5, similarity of matching texts to responses is high, therefore this metric makes the model desirable.

IX. CONCLUSION AND FUTURE SCOPE

Monitoring customer sentiments for real-time customer experience feedback is a crucial task for planning and maintaining their business models for competitive success and survival.The research experiment used Twitter mining and community sentiment analysis to fetch real-time sentiments of the masses. A positive sentiment score of a company is an indicator of the brand preference of the public and a negative sentiment score indicates customer dissatisfaction or inclination towards any other company which is better suited to their requirements.

With the addition of a dashboard as well as chatbot connected to the dashboard, we bring in a different dimension in viewing social media sentiments of different organizations, with the ability for business stakeholders to view real time trends, and make decisions on the fly. Data mining and sentiment analysis techniques can be used by managers to take timely actions to predict and prevent customer churn.

In the future we plan to incorporate various other metrics related to data mined from social media, and also enhance our chatbot capabilities.

X. APPENDIX

The below screenshots will represent some of the functionalities and the aspects of the dashboard.

Fig. 10. Dashboard Home Page

Fig. 11. Timeline of Latest Tweets for the Network

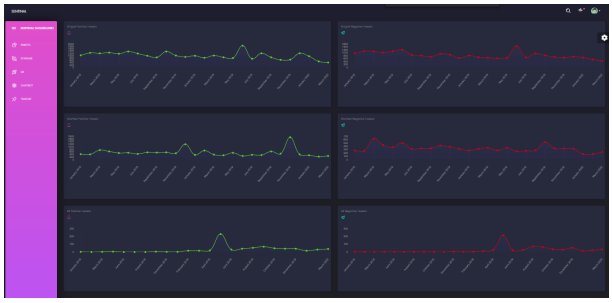

Fig. 12. Timeline of Number of Tweets

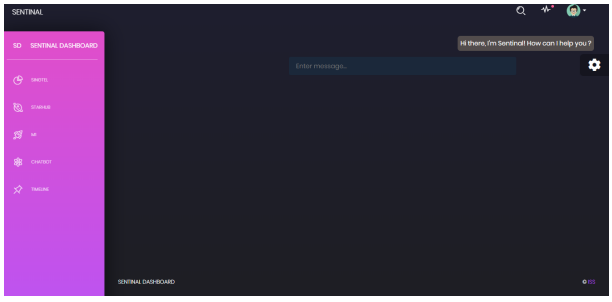

Fig. 13. Telecom Sentinal Chatbot

REFERENCES

[1] J. Poushter, "Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies," Pew Research Center, pp. 1–5, 2016.

[2] S. Ranjan and S. Sood, "Exploring Twitter for Large Data Analysis," International Journal of Advance Research in Computer Science and Software Engineering, vol. 6, no. 7, pp. 325–330, 2016.

[3] Y. M. Li and Y. L. Shiu, "A diffusion mechanism for social advertising over microblogs," Decision Support System, vol. 54, no. 1, pp. 9–22, 2012.

[4] I. Taxidou and P. Fischer, "Realtime analysis of information diffusion in social media," Proc. VLDB Endowment, vol. 6, no. 12, pp. 1416–1421, 2013.

[5] L. Harris and A. Rae, "Social networks: The future of marketing for small business," Journal of Business Strategy, vol. 30, no. 5, pp. 24–31, 2009.

[6] S. K. Janane, M. S. Keerthana, and B. Subbulakshmi, "Hybrid Classification for Sentiment Analysis of Movie Reviews," International Journal of Engineering Sciences and Research, vol. 7, no. 4, pp. 724–728, 2018.

[7] W. X. Zhao et al., "Comparing Twitter and Traditional Media Using Topic Models," in Proc. European Conf. on Information Retrieval, 2011, pp. 338–349.

[8] S. Kumar, F. Morstatter, and H. Liu, "Twitter Data Analytics," Springer, p. 89, 2013.

[9] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," Ain Shams Eng. J., vol. 5, no. 4, pp. 1093–1113, 2014.

[10] T. Wilson, J. Wiebe, and P. Hoffman, "Recognizing contextual polarity in phrase level sentiment analysis," Acl, vol. 7, no. 5, pp. 12–21, 2005.

[11] M. M. Mostafa, "An emotional polarity analysis of consumers' airline service tweets," Soc. Netw. Anal. Min., vol. 3, no. 3, pp. 635–649, Sep. 2013.