FIFA Twitter Real-time Sentiment Analysis

Abstract

Bidirectional Encoder Representations from Transformers (BERT) has become the state-of-the-art baseline for Natural Language Processing (NLP). Despite the sheer advantages of $BERT_{LARGE}$ in model performance, $BERT_{BASE}$ or even smaller BERT models are widely applied in the industry, likely due to their less strict hardware requirements, especially GPU Memory.

In this project, the team aims at using distributed training strategy on local devices to train a $BERT_{LARGE}$ model for sentiment analysis tasks. After comparing with various other classification models, we propose to deploy the best model onto PySpark Streaming as well as Tweepy to predict the sentiment implied in Twitter real-time and historical data.

Multiple interesting patterns are discovered, corresponding to critical events happening during FIFA World Cup 2022, and these results are presented with an interactive webpage constructed through Python Flask framework.

1. Introduction

Twitter is a popular online social media platform for users to share their immediate feelings and thoughts within a comparably shorter length of paragraphs. With the explosion of available information in the Big Data era, it is a general trend that understanding the opinion – or specifically, the emotions of the public – is gradually gaining its significance. The project team is primarily motivated to use modern Natural Language Processing (NLP) techniques to gain insights of English Twitter users' response regarding the ongoing 2022 FIFA World Cup.

In this work, our team first employs traditional NLP methods (e.g., word frequency, Python NLTK Vader Sentiment Analyzer, etc.) on a static, labeled dataset of Twitter users' message posts (i.e., tweets) and their corresponding sentiment of either positive (1) or negative (0). We then train multiple classification models – including Long Short-Term Memory (LSTM), Bidirectional Encoder Representations from Transformers (BERT), and $\text{BERT}_{\text{LARGE}}$ – on the static data to predict the sentiment of tweets, among which the pre-trained $\text{BERT}_{\text{LARGE}}$ model achieves the best performance on validation set, after using Multi-Worker Mirrored Strategy training. After training, we broadcast the weights of the saved, suitable BERT model both locally and onto PySpark, in order to predict the historical and real-time text data collected through Tweepy.

To adapt and evaluate our model in a topic with popular and impactful insights, we choose to focus on the analysis on the trending topic 2022 FIFA World Cup. Twitter API is used to extract both streaming and history tweets, during which hashtag (e.g. #Argentina, #Neymar), keywords filters, and other attributes (e.g., the time a tweet is posted, the language of a tweet, etc.) are applied to aggregate people's sentiments on a specific country or player. The results only provide insights on people's reactions to unexpected events, but also reveal the performance of the model based on the events happened during the FIFA tournament. In order to better visualize the predictions, we constructed an interactive HTML webpage with Flask and Plotly Python packages, where clients can choose among several popular countries to see the trends of public sentiment to these FIFA teams varying with respect to time.

As experiments to further ensure the model performance on soccer-specific tweets, we also manually label 16 positive data and 14 negative data during the streaming process, whose accuracy turns out to be 96.67%. Robustness tests including intentionally manipulating the texts are also performed to ensure that the model behavior is not sensitive to specific words or phrases.

The source code of this project is available on GitHub https://github.com/Sapphirine/202212-18-FIFA-sentiment-snalysis.

2. Related Work

2.1. Related Sentiment Analysis with Streaming Data

Studies similar to ours [9] have investigated real-time sentiment analysis on politics-related tweets mainly using Spark streaming, while also introducing Kafka to arrange

the streaming data from twitter API before sending to Spark Stream which could solve the bottleneck of processing the text analysis on a high-velocity information. It would be a good method if we want to pursue an efficient analysis framework. On the other hand, the proposed NLP method to do sentiment analysis is somewhat old-fashioned. Basically, the model classified the tweets into positive or negative by summing up the positiveness of each word in the tweet, and there are positive and negative dictionaries where all the words were stored. This rule-based analysis could be misleading due to the fact that no negation handling and modifier management is adapted to the sentiment classification [1].

2.2. NLP in Deep Learning Era

Witnessing the emergence of modern Artificial Intelligence, Recurrent Neural Network had long been the state-of-the-art method for NLP tasks, because of its memory property when dealing with sequential data. Long Short-Term Memory [5] model, by addressing RNN's gradient vanishing issue in long sequences through forget gates, further improved its capability, especially in the sequence-to-sequence tasks like machine translation.

Yet researchers later discovered that embedding words in a sequential, left-to-right order does not reasonably represent the inherent structure of human languages. In addition, RNN-based models work poorly in parallelization, as the input of later cells is dependent on the previous cells in a layer. Thus, the Self-Attention mechanism is proposed with the Transformer [7] model. It is worth admitting that Transformers, by every means, marked a cutting-edge milestone for NLP; meanwhile, its transductive, encoder-decoder architecture – originally designed for translation tasks – encountered drawbacks when generalizing to text classification tasks like sentiment analysis.

Based on self-attention in the previous approach, the direct descendant Bidirectional Encoder Representations from Transformers (BERT [2]) forwards both within-sentence word attentions and cross-sentence correlations, in a broader scope compared with traditional Transformers. Even better for our specific task, BERT also introduced a self-supervised learning strategy that makes transfer learning applicable – the pre-trained parameters are label-independent and thus more generalizable.

2.2.1 $\text{BERT}_{\text{BASE}}$

Devlin et al. [2] initially proposed the model with 2 different levels of complexity, among which $\text{BERT}_{\text{BASE}}$ has $L = 12$ layers, hidden size of $H = 768$, and $A = 12$ self-attention heads (approximately 110 million parameters in total). These structural hyperparameters were chosen to mimic the model size of, and thus compare with Generative Pre-trained Transformers (OpenAI GPT; Radford et al., 2018). The $\text{BERT}_{\text{BASE}}$ model, as well as other smaller variants like $\text{BERT}_{\text{MEDIUM}}$ or $\text{BERT}_{\text{SMALL}}$, has been widely adopted in various text classification tasks, especially sentiment analysis ones [6] [8].

2.2.2 $\text{BERT}_{\text{LARGE}}$

The primary purpose of $\text{BERT}_{\text{LARGE}}$ is to challenge the highest potential performance of this model. The authors set hyperparameters of $L = 24$ layers, $H = 1024$ hidden size, and $A = 12$ self-attention heads, resulting in 3 times more parameters, compared with $\text{BERT}_{\text{BASE}}$.

Such increase in performance comes with cost – in practice, training a $\text{BERT}_{\text{LARGE}}$ model typically requires at least 16 GB of GPU Memory, which is oftentimes beyond the capacity of a single GPU worker. In the industry, researchers [8] expecting higher performance of $\text{BERT}_{\text{LARGE}}$ models may compromise to use more concise ones, due to the limitation of computational power.

3. Data

3.1. STS Data

The static dataset we use is *Stanford Twitter Sentiment (STS)* [4] which includes 1.6 million tweets and their corresponding sentiments. Other trivial information like User ID, Tweet ID, and post date are also included in the dataset. Initial explorations show that each tweet has 74.09 alphanumeric characters, from which 16.41 words are formed, on average. Further, instead of manually labeling data, the sentiments are generated in a seemingly silly but efficient and highly accurate way – researchers streamed through all tweets in a certain period of time, filtered out all tweets with emoticons consist of special characters ":)" and ":(", and labeled them as positive and negative, respectively.

One of the big challenges of the dataset is that BERT requires a large scale data due to its gigantic model size. This is the main reason we choose to use STS instead of fine-grained dataset and emotion detection dataset as both of them needs to be manually labeled to ensure their accuracy. If the label is automatically generated by models, it is still not feasible to fit BERT on the model-based dataset. Thus, the safest way is to use binary classification data which could acheive both high accuracy and efficiency. And since it is hard to extract the streaming data from twitter API within limited time, using STS is the best choice for our project, which has also been broadly used in the research such as twitter sentiment analysis and subjectivity analysis [3].

3.2. STS Data Preprocessing and Exploration

Due to the sparse nature of human natural languages, data preprocessing procedures can be challenging. The project team first drops trivial columns, and excludes all information irrelevant to sentiments through Regular Expression. For example, mentioning other users in a tweet, in the format of "@SomeUserID", as well as URLs starting with "http://" and "https://", are all removed, with the belief that the characters in User IDs and URLs will not influence the sentiment when composing the tweet. In the raw dataset, sentiments are assigned labels of 0 (negative) and 4 (positive), which are transformed to 0 and 1 in the cleaned dataset.

Next, we apply word tokenizer in Python NLTK to split each text into words, bigrams, and trigrams, and count their frequency of occurrences in both positive and negative tweets separately. The odds of a certain element (a single word, 2-word, or 3-word phrase) belonging to a positive tweet against that belonging to a negative one are visualized in the scatterplot below. The diagonal represents the line on which elements have equal probability of belonging to positive and negative, and the more an element deviates from the diagonal, the more extreme in sentiment it can be interpreted. Result shows that words generally have the most stark positive-negative contrast.

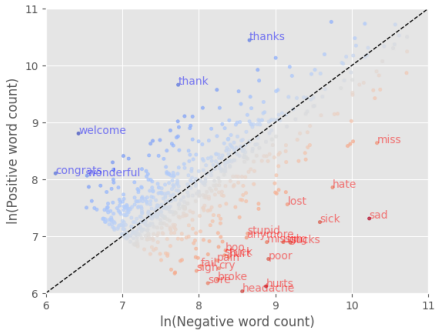

Figure 1. Word Frequency with positive and negative analysis

As a reference, the Vader Sentiment Intensity Analyzer is also applied to the texts in the dataset to visualize the distribution of positive and negative emotions. The model generates a continuous, context-independent Vader Score between -1 (negative) and 1 (positive). The dataset is then sorted in the ascending order of Vader Score, with each red and blue point representing tweets with negative and positive labels. It is worth emphasizing that the Vader model tends to make 0 predictions when the text seems to be neutral, while the labels in the dataset are dichotomous.

3.3. Collected Twitter Data

The scope of this project shall not be limited to existing STS dataset, as it is widely accepted that only trending results can arouse business interests and real-world impacts. As the project is moving on, the trending topic of FIFA World Cup comes into our sight. Based on intuition of real-life experience, the project team believes that the tweets by soccer fans are oftentimes associated with strong and stark-contrasted emotions. For example, users may express their happiness when the team they support wins a match, or express their pity (or even anger) when the team loses. In addition, during a certain popular match, it is expected that there will be a mix of cheering (positive sentiments) and anxiety (negative sentiments) in the corresponding topic.

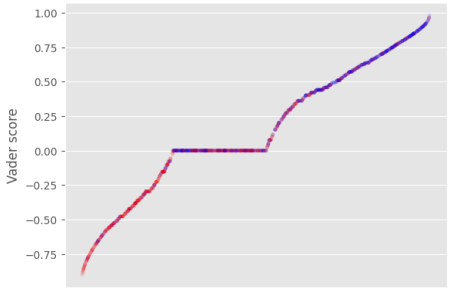

Figure 2. Vadar Score

Thus, it is reasonable to conclude that current Twitter data about FIFA World Cup will likely trigger polarized sentiments of the public, and as a result, potentially bring about useful insights like the support rate of a certain team. In a practical term, the team proceeds to collect data in the following aspects.

3.3.1 Real-time Streaming Data

Real-time data is an immediate reflection of Twitter users' opinion with regards to the selected topic. On December 16th, we collected streaming data using Tweepy for 3 hours in total. The search query selected was that tweets that includes BOTH (1) "#fifa" or "#worldcup2022", AND (2) "#argentina" or "#france" will be returned. (Note that Argentina and France are the two soccer teams which were about to compete in the FIFA Final on December 18th, so upon the day the data is streamed, these two hashtags are possibly associated with people's feelings and attitudes towards these teams.)

Other features of the streaming data includes:

- The average velocity of the streaming data is approximately 44.7 tweets per minute.
- The information collected includes (1) body text of a tweet, and (2) the time when the tweet is posted (or streamed by Tweepy), with an interval of 5 seconds. For example, a tweet posted on 15:11:08 (HH:MM:SS) will be logged as precise as 15:11:05.

It is worth notifying that, in consideration that future replication of the real-time analyses may be required, as well as the fact that treating tweets sentence by sentence may be inefficient, compared with parallel operations, we further decide to handle the streamed data using PySpark DStream format to aggregate the data with time series into static RDD, and then save them into storage (CSV file).

Figure 3. Sample Streamed Data

The above is a snippet of 5 raw tweets that involves Argentina. The full document can be accessed by "dataset/data_stream.csv" in the GitHub repository. We can see that the majority of the tweets are considered positive sentiments.

On the other hand, similar to the previous STS dataset, the streaming data is even more messy. Besides Usernames and URLs which has been treated as irrelevant to sentiment, the newly collected data even contains a lot of emojis (e.g., the sign of a soccer ball). These non-ASCII characters will introduce huge difficulty during the model training phase, so the team adopted a brute force approach to decode the text as ASCII format, and dropped all characters that cannot be decoded.

Further, Twitter users tend to add blank lines between their short paragraphs, which will make the format unclear and harder to tokenize. Thus, we replace all new line characters (i.e., "\n") with white spaces " ".

3.3.2 Historical Static Data

Due to the time constraints, it is impossible for the project team to collect the tweets in every insightful moment merely through streaming on PySpark and Google Cloud Platform. Fortunately, Twitter also provides API for developers to search recent tweets, if not distantly historical ones.

Tweepy Search API's are comparably more powerful than streaming function. Specifically, the search results can return the following attributes of a tweet:

- The user ID who posted this tweet
- The body text of a tweet
- The time when the tweet is posted
- The geographic location (in encoded IDs) of the user associated to this tweet
- The language (specified by Twitter) in which the tweet is composed
- Whether this tweet has any replies or retweets, and if so, the above information of the retweets
- .....

With all these features available, the team members lay much emphasis on the static results as the major source of data. Tweets are queried by the timeline, hashtags, words or phrases in the body text, and its language. Specifically, the data of English tweets related to one or combinations of following categories are collected:

- FIFA: e.g., #fifa, #worldcup, #fifa2022, etc.
- Country names (whose teams are popular): e.g., #argentina, #brazil, #england, #france, #spain, #portugal, etc.
- Last names of superstar players in an ongoing match: e.g., (Leandro) Paredes, (Lionel) Messi, (Lautaro) Martinez (from Argentina), (Denzel) Dumfries, (Virgil) van Dijk (from the Netherlands), etc.

With the Elevated Access of Twitter developer account, we have retrieved historical data as early as November 21st, and the total volume of data collected is more than 50,000, including the streamed data.

Yet there has still been a shortcoming of Search API – within each request it can only return at most 100 tweets. With this upper bound restricting, we have to make requests for tweets within a very short time window to avoid losing too much information because the number of tweets exceeds 100 during that relatively longer time window. In practice, a typical choice of window size, if the total time span is not too large, is 1 hour (which allows the maximum of 100 tweets per hour).

It is intuitive that the real-time streaming data and historical static data indeed come from the same distribution. Due to their similar nature, the data cleaning techniques are also directly applied to these collected data.

4. Methods

4.1. LSTM

Easy to implement and fast to train (comparing with obviously more complex models), Long Short-Term Memory model is considered a baseline by the team members.

Admittedly, this is not a typical baseline approach, as some may argue that there are more traditional and even simpler model – say, Naïve Bayes – for sentiment analysis,

or more generally, binary text classification tasks. Yet in our opinion, Naïve Bayes model, with the following formula as theoretical foundation, has several shortcomings which undermines, if not totally denies, its use in our topic.

$$\begin{aligned} p(c|x_1, x_2, ..., x_n) &\propto p(c, x_1, ..., x_n) \\ &\propto p(c)p(x_1|c)p(x_2|c)...p(x_n|c) \\ &\propto p(c)\prod_{i=1}^{n} p(x_i|c) \end{aligned}$$

The shortcomings of Naïve Bayes model include:

1. NB is solely dependent on how $x_1, ..., x_n$ are chosen, or even more importantly, the dictionary size n. In our context, however, tweets, especially those posted during or after an exciting game, are oftentimes not formal sentences with rigorous lexical and grammatical structure. For example, there are tweets consists of nothing but cheering and exclamations. This abnormal, sparse pattern of word choices will make it harder for NB model to associate sentiments to tweet texts.

2. NB assumes that the distribution of x_i is independent of $x_j, \forall i, j \in 1, ..., n$. But in modern era in NLP, consensus are that the relationships between words *are* indeed significant and should be modeled effectively.

The choice of LSTM model can, in return, alleviate these issues. After proper hyperparameter tuning, the team decides to encode the data using a vocabulary size of 5,000, and embed each sentence as a vector of length 64. To override the unwanted feature that LSTM treats a sentence (sequence of words) in its inherent order (i.e., from left to right in sentences), we used 2 adjacent Bidirectional LSTM (or so-called BiLSTM) layer in our model, each with 64 outputs channels. After those, two dense layers of 64 neurons and ReLU activation are added on top, followed by the last output layer of 1 neuron and Sigmoid activation.

The model is trained for 5 epochs, with Adam optimizer of learning rate 0.001. We can see that the baseline model achieves a very promising validation accuracy of 82% merely in the first epoch. Yet it already starts converging, and the performance aligns to 82% in the rest of the training session. Considering the relatively long input sequences of strings, it is reasonable that the long-term memory feature enabled by Forget Gates are less capable than the attention mechanisms. Thus, we do not expect perfect performance of such model.

The 5-epoch training session takes 2 hours on our local device. It is worth admitting that it takes longer than we expected, probably due to the sequential nature of RNN-based models – i.e., the input of next cell must wait for the output of previous cell before it is activated – which does not support parallelism on GPU. Such inefficiency motivates the team members to explore more complex approaches.

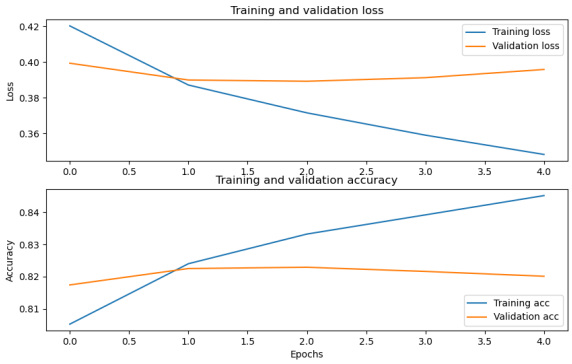

Figure 4. LSTM Accuracy and Loss Comparison

4.2. BERT$_{\text{BASE}}$

Inspired by the superiority of Attention mechanism, compared with Forget Gates feature, the team first explored Transformers. During our research, however, we discovered that Transformer model may not be the best fit for our sentiment analysis, specifically in the following aspects:

1. Even though Transformer can process text data in a parallel and efficient manner, the model is still "unidirectional", which means that the sequence is treated from left to right in its input order. Admittedly, this feature is acceptable or even desired in some cases, when performing sequence-to-sequence tasks like machine translation, for which Transformer is originally designed. Yet in our sequence-to-one text classification task, it is more important for the chosen model to treat the full sentence in an integral manner, with even more emphasis laid on the relationship between sentences, given that many sentences in the tweets are relatively short and fragmental in our context.

2. The encoder-decoder structure of Transformer introduces bottlenecks to training, since the decoder phase is both insufficient and unnecessary, unless sequential output is needed. The collected 1.6 million dataset is large at the first glance, but the sparsity of text in tweets is also an outstanding obstacle, not to mention the potential difference in distribution of STS Twitter data and real-life Twitter data. For this reason, it is doubtful whether Transformers will have ideal performance on validation data, or even the real-life data upon future model deployment. Thus, we tend to avoid wasting computational power in inefficient and cost-ineffective model training.

3. Due to its supervised-learning nature, a Transformer model must train from scratch for each specific task and corresponding dataset. In the area of NLP, however, the best approach is to enable Transfer Learning from very large source data of general human language patterns, while the information implied within the relatively smaller target data should be made the best use of, aiming at the goal task. In fact, there exists other models like BERT which is pre-trained on large unannotated data (e.g., Wikipedia and English books), and can thus be fine-tuned to our Twitter sentiment analysis task.

Upon studying multiple Deep Learning models in NLP, the project team decides to apply a pre-trained, uncased $\text{BERT}_{\text{BASE}}$ model obtained through TensorFlow Hub. To perform transfer learning, another dense layer with 32 neurons and ReLU activation is added on top of BERT, followed by an output layer of 1 neuron with Sigmoid activation function.

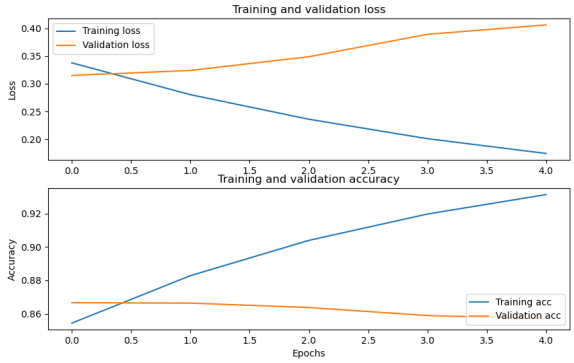

Figure 5. BERT-base Accuracy and Loss Comparison

During our experiment on the team's local devices, a 5-epoch training session of $\text{BERT}_{\text{BASE}}$ takes less than 17 hours on a single GeForce RTX 3080 GPU with 10 GB memory. With proper hyperparameter tuning, this model achieves a Training Accuracy of 94.16% and Validation Accuracy of 85.70%. The corresponding best set of hyperparameters are: batch size = 32, learning rate $= 3 \times 10^{-5}$ for Adam optimizer. At this point, we tend to conclude that the result is promising, and that it well reflects BERT's capability in deep understanding of the context and relationships between words in a sentence.

Regardless, it is worth pointing out that despite its high accuracy, the automatically generated labels in STS dataset are, after all, not ground truths. That says, minor inaccurate predictions from classification models should be allowed.

4.3. $\text{BERT}_{\text{LARGE}}$

The project team further proposes to experiment with more complex models, specifically the $\text{BERT}_{\text{LARGE}}$ model with 340 million parameters. Considering that the model size (16GB) already exceeds the GPU memory of every available hardware (10 GB or 8 GB) alone, we employ a Multi-Worker Mirrored Strategy for distributed training.

Different from simple and traditional Mirrored training strategy, where a copy of the full model parameters is kept in both workers and mirrors each other when performing gradient descent, in Multi-Worker Mirrored Strategy, only part of the full model is loaded on each local worker, and the partitioned models will communicate the model parameters in every step of the training session. We initiate a local server to connect our machines, and utilize Nvidia NCCL implementations (through TensorFlow Distributed Strategy methods) to achieve Collective Communication between the GPUs.

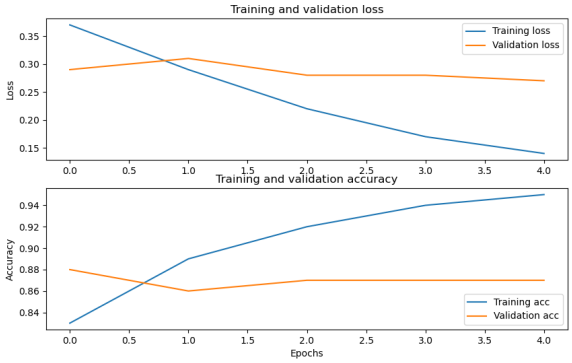

Figure 6. BERT-large Accuracy and Loss Comparison

Doing hyperparamter tuning of such a complex model is disastrous on local server, so we just used almost the same set of hyperparametes of $\text{BERT}_{\text{BASE}}$, except that the original batch size of 32 is changed to a global batch size of 64, where each of the two workers takes in 32 data instances in a batch at one time. We can see from the above plot that the training accuracy of $\text{BERT}_{\text{LARGE}}$ is initially slightly lower than that of $\text{BERT}_{\text{BASE}}$, but exceeds later as the training session goes on. This is within the expectation of team members, as it is reasonable for models with more parameters to take more time to converge.

The validation accuracy of $\text{BERT}_{\text{LARGE}}$ after 5 epochs is 87.02%, as compared to 85.70% of $\text{BERT}_{\text{BASE}}$. Despite its slightly higher accuracy, the huge complexity and associated high computation power demanded cast doubt on the cost-effectiveness of $\text{BERT}_{\text{LARGE}}$ model. The team decides that the 1.32% increase is an unfair tradeoff, and that we should move on with $\text{BERT}_{\text{BASE}}$.

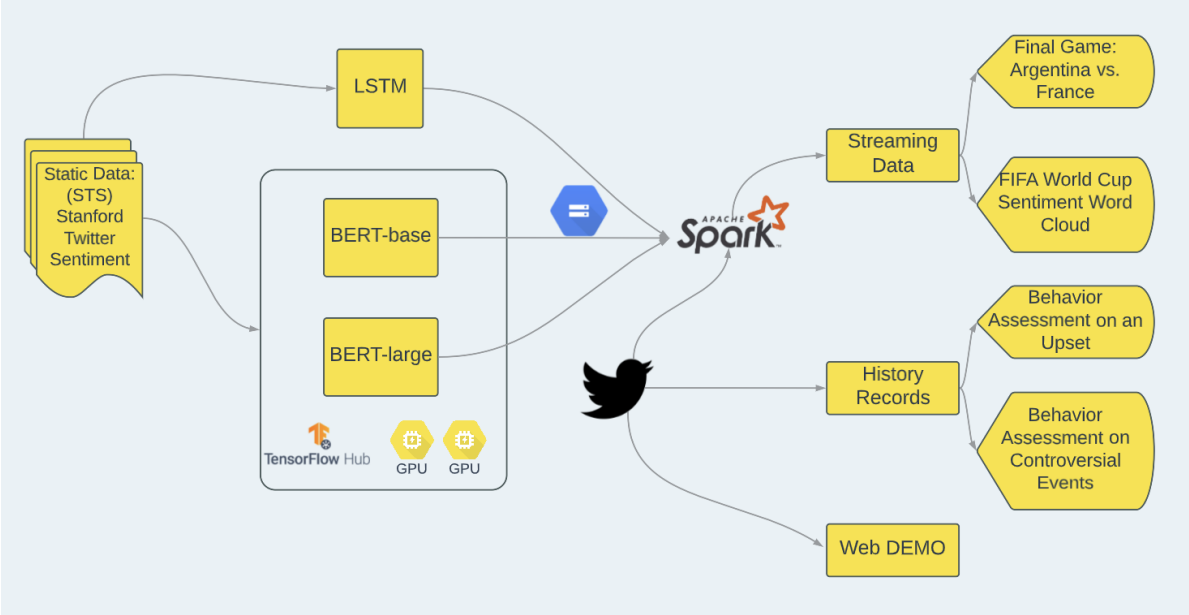

Figure 7. System Architecture Overview

5. System

5.1. Architecture

Figure 4.3 shows the architecture of our project. The database our project use to train the models is called Stanford Twitter Sentiment which is proved to be reliable for building sentiment analysis models as mentioned previously. Next, we proposed different NLP models such as LSTM, $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, where significantly related packages such as Tensorflow, Tensorflow Hub, and the Multi-Worker Mirrored Strategy are adopted. We store the models to Google Cloud for future deployment. Then, we acquired online dataset from Twitter API. The datasets include both streaming data and history data. Spark is used to help extract the streaming data, while we extract history records from local. We make predictions on those data and visualize them in various ways using mainly Matplotlib and InfraNodus. We also create a demo to be an interactive web page that the client can play with. The demo basically involves Flask and Plotly.

One of the potential bottlenecks the overall system has is that the training data is too general compared to the specificity of our research topic. It could be better if the training dataset is also topic-specific, however, there is always a trade-off between data volume and topic-specific. Also, FIFA data contains lots of emojis, exclamation, and most of the tweets have no rigorous grammatic structures, while the training data is cleaned. Secondly, since we only predict the tweets that are in English, this will miss lots of valuable information that is in other languages. This is especially important because FIFA is an international sports events. Thirdly, twitter API only provides data from the previous 30 days, and the requests amount is too limited to extract enough data even after account elevation, so we could neither effectively do the sentiment analysis from the beginning of the tournament nor analyzing all the countries we are interested in.

5.2. Software Stack

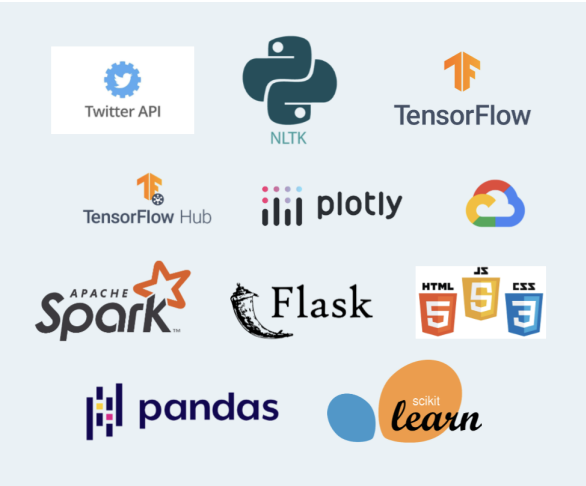

Figure 8. Software Stack

Figure 5.2 shows the various softwares and packages we use during the system building process. We use libraries NLTK, TensorFlow, Scikit-learn, TensorFlow Hub to train the model, and pandas as a coordination. We then use google cloud platform to operate Spark to help us manipu-

late data streams generated from twitter API. Lastly, we use several visualization tools to present our results. For the visualization part, we typically choose plotly, matplotlib and infranodus to plot important finds in our research. Flask is used as the web framework for the project, providing a lightweight and easy-to-use interface for building the web application. Python is chosen as the main programming language for the project due to its versatility and wide range of libraries for data analysis and deep learning. The project is developed on a Windows System, and tested on GCP Linux (Debian). The demo is presented on macOS.

5.3. Results

Our major results with meaningful insights are primarily presented with graph visualizations, because the team needs to carefully handle the selection and customization of the visualization, with the purpose of clearly showing the patterns and insightfully communicate the results to the audience. The visualizations on the interactive webpage, on the other hand, are in a uniform manner, and thus serve as a general overview.

In the context of the FIFA World Cup 2022, sentiment analysis of tweets about the event could potentially provide insight into how people are feeling about the tournament, the teams and players involved, and any related issues or controversies. Here are a few interesting research questions we think that Twitter sentiment analysis could potentially provide for the FIFA World Cup 2022. Note that all of the predictions are based on model $\text{BERT}_{\text{BASE}}$.

- As the final game approaching, which final team people on Twitter feel more positive? Argentina or France?

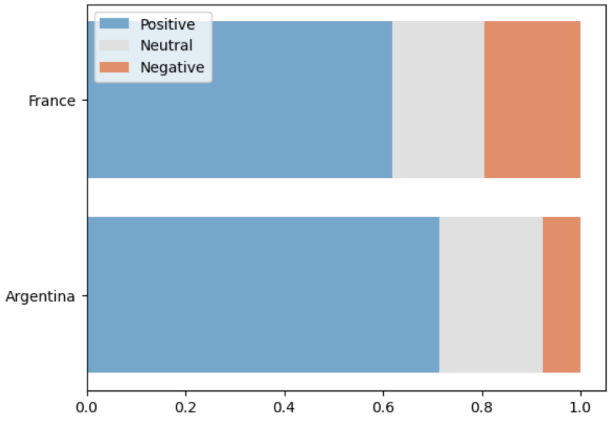

Figure 9. Final Game Teams Sentiment Before Final

Figure 5.3 shows that Argentina receives relatively higher positive reviews than France on Dec 14 before the final begins. The public sentiments are mostly positive which is intuitive.

- Sentiment about specific teams and players: As Morocco defeating several top teams in the previous games, this team becomes unexpectedly popular under the public. On the other side, as the champion of FIFA 2018, France is also constantly attracting attention from all over the world. We are interested in which team is more popular over the seven days covering their knockout match?

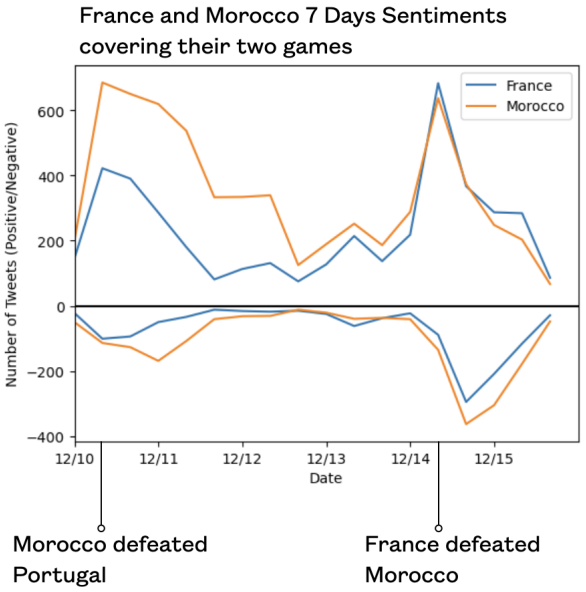

Figure 10. France and Morocco 7 Days Sentiments Covering their games

From the figure above, we can clearly see two different peaks in the plot, one around December 10th , the other around December 14th. On December 10th, both teams had there quarterfinal games. Morocco defeated Portugal and became the first Arab and African country to make it to the round of four in the tournament history. This was unexpected and was a big upset. You can see how people were positively discussing about Morocco. On the other hand, France defeated England and moved on to semifinal to play against Morocco. In the following days, people have discussed about both teams while more comments were about Morocco. Finally when the two team met in the semifinal on December 14th, the positive and negative discussion about the two teams once again increased dramatically as this plot clearly shows.

- Sentiment about specific issues or controversies: Are those critical events between Argentina and Netherlands generating a lot of discussion on Twitter? How are people reacting to these issues?

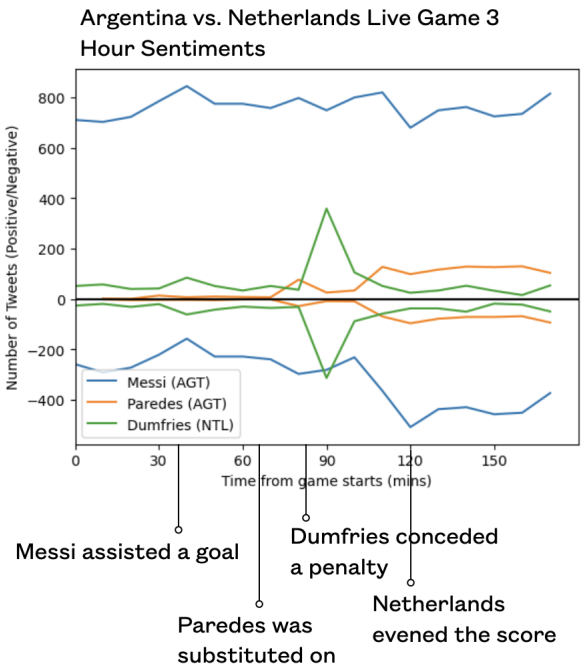

Figure 11. Argentina vs. Netherlands Live Game 3 Hours Sentiments

On December 9th, during the match between Argentina and Netherlands, both players inside the field and on the substitutes bench from both teams involved in pushing and hustling. This plot above shows the sentiment analysis over three main players. A critical point happened in the 73th minutes when Dumfries brought down Acuna inside the box and conceded a penalty to Argentina. At that time, both positive and negative trends on Dumfries has increased dramatically as fans from both side were discussing about the poor challenge by him.

5.4. Demo

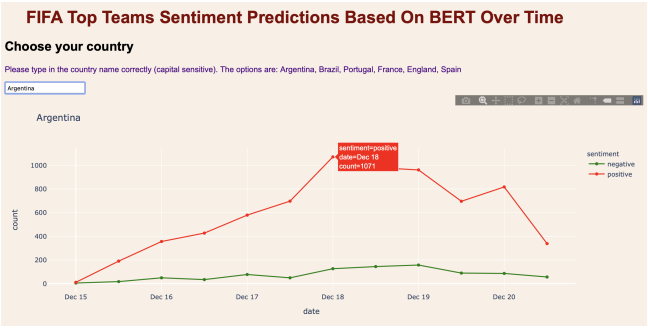

Figure 12. Demo

Our demo mainly provides the client a FIFA 2022 top team sentiment predictions during the weeks surrounding the final match. The predictions is generated from the implemented $\text{BERT}_{\text{BASE}}$ model. A line chart with positive and negative lines show the change of sentiments over time. An input area is rendered above so that the user could query different countries. When a country name is entered, a request would be sent to route and the plot would be updated. The user could use the Plotly built-in functions such as zoom-in, screenshots.

6. Experiments

6.1. Model Evaluation Metrics

Besides overall accuracy, other metrics including Precision and Recall are also applied to evaluate the model performance, where $\text{BERT}_{\text{LARGE}}$ always has the highest performance.

Table 1. Models Comparison with Different Performance Metrics

Methods	Accuracy	Precision	Recall
LSTM	82.01	81.72	82.39
$\text{BERT}_{\text{BASE}}$	85.70	85.44	86.15
$\text{BERT}_{\text{LARGE}}$	87.02	86.80	87.36

Further, given that the dataset if very balanced (number of positive labels approximately equals that of negative labels), it is reasonable that precision and recall, as defined below, are close to the overall accuracy.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

The PR-Curve and ROCare shown below 6.1. We can see that they all have smooth, typical shapes showing tradeoffs.

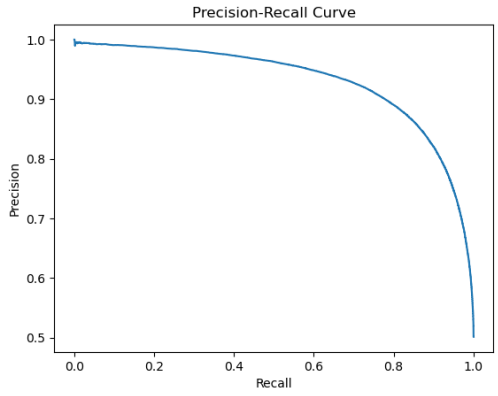

Figure 13. Precision-Recall Curve

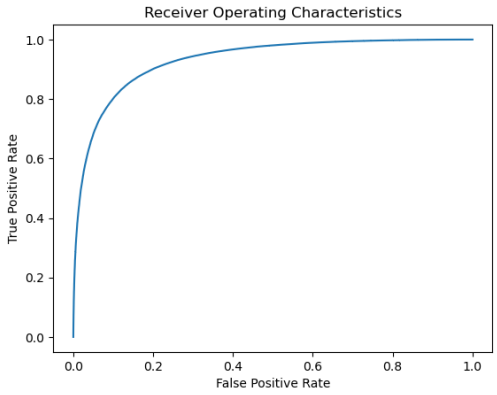

Figure 14. Receiver Operating Characteristics

6.2. Performance Experiments

Besides evaluating on validation set, the robustness of the selected model can be further tested in a self-supervised manner, by intentionally manipulating the text data and comparing its new prediction with the original one.

One common experiment for robustness is the negation test, during which a strong negation word is manually inserted into each sentence in the test set. It is expected that the model should generate an opposite outcome. For example, since "The food is good" is considered as positive, the model should, if well mimicking human intuition, assign a negative label to the opposite sentence "The food is not good."

On the other hand, it is also considered a desired behavior for a model if, without explicitly adding strong positive or negative words or phrases, the prediction of sentiment should be robust to the minor changes in irrelevant information in the text. For example, "Issac Newton likes the mild weather in London" should reasonably has the same label with "Albert Einstein likes the mild weather in New Jersey", under the common consensus that people's names and locations are oftentimes irrelevant to the sentiment of a sentence.

We constructed both types of robustness test of 10 cases each, and the $\text{BERT}_{\text{BASE}}$ successfully passes through all the test cases. The 100% passing rate demostrates the ability to understand the context and cross-word relationship of BERT model.

7. Conclusion

In this work, we presented a full pipeline of Big Data Analytics methods, which includes in-depth theories of Neural Network and Artificial Intelligence, distributed training of partitioned models across devices through TensorFlow, data streaming and model broadcasting through PySpark, and front-end webpage implementation of interactive visualizations through Flask.

Our analysis reveals the strong correlation between the ongoing events and the changes in Twitter users' real-time sentiment. We are excited to see the potential of our system (or similarly constructed systems) in the application in other areas of study, especially om the field of sports analytics like NBA or NCAA (American Football), or in other languages than English.

One of the potential future improvements could be to further perform Emotion Detection tasks on top of binary Sentiment Analysis. Understanding anger, fearness, etc., instead of a general "negative" sentiment can likely bring about more advanced and detailed insights.

References

[1] Muhammad Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad, Maria Qasim, and Imran Ali Khan. Lexicon-enhanced sentiment analysis framework using rule-based classification scheme. *PLOS ONE*, 12(2):1–22, 02 2017. 2

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. 2

[3] Anastasia Giachanou and Fabio Crestani. Like it or not: A survey of twitter sentiment analysis methods. *ACM Comput. Surv.*, 49(2), jun 2016. 2

[4] Alec Go. Sentiment classification using distant supervision. 2009. 2

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 2

[6] Manish Munikar, Sushil Shakya, and Aakash Shrestha. Fine-grained sentiment classification using bert. In *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, volume 1, pages 1–5, 2019. 2

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 2

[8] Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. Bert post-training for review reading comprehension and aspect-based sentiment analysis, 2019. 2

[9] Nashwan Zaki, Nada Hashim, Yasmin Mohialden, Mostafa Mohammed, Tole Sutikno, and Ahmed Ali. A real-time big data sentiment analysis for iraqi tweets using spark streaming. *Bulletin of Electrical Engineering and Informatics*, 9(4):1411–1419, 2020. 1

8. Appendix

8.1. Contribution

- Anne is responsible for system design, visualization, and Flask implementation.
- Robert is responsible for model training and deployment.
- Both team members contributed fairly equally to other works, e.g., data preprocessing, cloud streaming, etc.