SENTIMENT TIME-SERIES PREDICTION IN TWEETS WITH LSTM

Project Number : 37

הוגש במחלקה להנדסת תוכנה
המכללה האקדמית להנדסה ע״ש סמי שמעון

אישור המנחה ________________________

אישור ראש המחלקה______________________

תאריך עברי
סיון ה'תשע"ט

תאריך לועזי
יוני 2019

באר שבע

Contents

Abstract

Twitter, an online social networking platform which allows its users to send messages called ‘Tweets’. The hashtag (the # sign followed by a phrase to a tweet, for example, #trump) is probably the most important function of Twitter search, and the most used.

Sentiment Analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a topic is positive or negative. We have built a sentiment analysis based on Machine learning algorithm and NLP (natural language processing) to categorize real-time tweets as Positive or Negative.

In this project, we are focusing on predicting the next data points of a hashtag over the next hours based on the indication of analyzing the sentiment in each tweet.

Using categorized data allow us to build a model which predicts the next hours on a time-series using Deep learning method. A powerful type of neural network designed to handle sequence dependence (such as time-series) is called recurrent neural networks. The Long Short-Term Memory network or LSTM network is an extension for recurrent neural networks, which basically uses RNN with extended memory, and allow us to observe datapoints as a sequence, and predict the next points.

Keywords: Twitter, Hashtag, Sentiment Analysis, Natural Language Processing, Prediction, Recurrent neural network, Long Short-Term Memory, Time-Series.

Acknowledgements

We would like to thank our supervisor Dr Marina Litvak . Thank you for your help, continuous support throughout the project. Your dedication has provided great value, your ideas have helped us shape the intention of the project.

Introduction

These days people are using social media to share their opinions. Twitter has become a huge data source for collecting human opinions about many fields including politics, marketing , stocks and more. A hashtag (#) is a markup that lets users select a topic, hashtags are very popular among twitter users, therefore, it is very convenient to track a hashtag in order to analyze users' opinions.

One of the purposes of tracking a hashtag is the ability to read each of its tweet and build a model that understand its meaning. Sentiment Analysis is a popular field in social media which allow us to build a model that can understand the meaning of each word from a sentence. For example: Positive: "I love the weekends" , negative: "I hate the weekends".

Using sentiment Analysis let us analyze user's opinion and classify them as positive or negative.

Today, Artificial Intelligence has grown widely and deeper into understanding humans' brain. The term "artificial intelligence" is used to describe machines that mimic "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving". One of the AI fields is called Recurrent neural Network.

Recurrent means the output at the current time step becomes the input to the next time step. At each element of the sequence, the model considers not just the current input, but what it remembers about the preceding elements. The ability to predict results out of a sequence. This method refers to Prediction over time-series.

Time series analysis refers to the analysis of change in the trend of the data over a period of time. Time series analysis has a variety of applications. One such application is the prediction of the future of a topic based on its past values.

In the context of data mining, pattern recognition and machine learning time series analysis can be used for clustering, classification query by content, anomaly detection as well as forecasting.

Theory

1. Related work

Techniques for time series prediction that is based on sentiment analysis has been tested before in different approaches and different datasets.

[Anshul & Arpit ,2011] investigated the causative relation between public mood as measured from a large-scale collection of tweets and DIJA values. They aimed to predict stock prices using sentiment analysis on tweets. They have tested Linear regression, logistic regression, SVM and SOFNN. SOFNN (reinforcement learning) yields the highest accuracy of 75.56%.

[ASUR, S. AND HUBERMAN , 2010] aimed to predict the future in social media, they have constructed a linear regression model for predicting box-office revenues of movies in advance of their release, and the results has shown that there is a strong correlation between the amount of attention a given topic has and its ranking in the future.

[EDWIN LI, 2018] provided an approach in applying two different methods: binary classifier and regression on an LSTM for predicting stock prices. The results showed that binary classifier works well, but the regression model yields greater results.
Increasing the amount of data, both in quantity as well as in adding more features, consistently improves both models.

2. Data Preprocessing

Data preprocessing is an important step in data mining. Twitter language, as any other social media language, may be hard to understand due to misspelled words, sarcastic meaning and slangs. The purpose of data pre-processing is to clean and correct the data for our machine to analyze with higher accuracy.

3. Sentiment Analysis

Sentiment Analysis refers to the use of natural language processing which is sub-field of artificial intelligence that helps machines to deal with human languages, to identify, extract and study affective states and subjective information .
Sentiment Analysis helps data scientists to analyze any kind of data i.e., Business, Politics, Social Media etc.

3.1. Support Vector Machines (SVM)

SVM is a supervised machine learning algorithm that analyze data that is used for classification for predicting a label or for regression for predicting continuous value.
Supervised method is preferred to apply when a data and their respective infers statistics about the feature of the data is available.

4. Time-series

A time series is a series of indexed points on a graph listed in time order. Time series lets us review a subject through time, observing changes may help us in predicting the future. One major issue in time-series is missing values, which we must take as a consideration when analyzing time-series. Analyzing time-series can be divided into three main components: Trends, Seasonal, Variations and other cyclic changes. One of the main problems in time series data are trends, since trends are unexpected events, it is harder to observe and predict their future.

- **Stationary Time Series :** Time series are stationary if they do not have trend or seasonal effects, when a time series is stationary, it can be easier to model.
- **Non-Stationary Time Series :** Observations from a non-stationary time series show seasonal effects, trends, and other structures that depend on the time index.

4.1. Missing Values

In previous related works, it was mentioned that the most important aspect in time-series analysis and data is the ability to handle missing data. It is fine if the amount of missing data instances is not very high, but if it is high, then the average of the highest and lowest values is a better alternative.

4.2. Trends

The increasing or decreasing value in the series. Trend is an event in a time series that usually existed a long time ago, trends usually receive high attention and then their rank changes drastically, Therefore, it is impossible to use one technique for trends, since trends can be linear or exponential. The easiest way to compute a trend in a time series is to compute linear trend plus noise (errors), although, some prefer filtering out noises because they may interfere to identify the patterns. If the time series data contain considerable error, then the first step in the process of trend identification is smoothing.

4.2.1. Smoothing data – Moving average method

Replacing each element of the series by either the simple or weighted average of *n* surrounding elements, where *n* is the width of the smoothing "window" (Box & Jenkins, 1976; Velleman & Hoaglin, 1981. A moving average for a given time period is the average of the values in that time period and those close to it. The longer the moving average period, the stronger the smoothing effect and the shorter the smoothed series.

5. Sentiment Prediction in Time series

Sentiment prediction is one of the main tasks in time series mining is an important research topic. Time series is analyzing temporal data, where time-series mining is the process of mining a lot of data in a time-series.

Time series is always non-linear; Therefore, sentiment prediction is a problem. Any sentiment is occurrence which effects time-series at point T. Events can be instantaneous or can have a duration.
Sentiment prediction consist four main components:

1. Preprocessing – processing data for higher accuracy, better prediction. Where the main components are feature extraction and labeling.
2. Model derivation – using previous time-series data in order to predict the future.
3. Sentiment prediction – predicting future sentiments of users, according to derivation model.

4. Sentiment Detection – sentiments are detected based on the predicted time-series and available patterns. Predicting a sentiment can be accomplished in two different approaches, Machine learning methods or Statistical learning methods.

5.1. Recurrent Neural Network

RNNs are feedforward neural networks augmented by the inclusion of edges that span adjacent time steps. An RNN has the ability to selectively pass information across sequence steps, while processing sequential data one element at a time. One of the weakness of RNN is the ability to view previous data point and predict the next point, in time-series , we aim to predict the dependencies at each time point, therefore, we need a network that can rely on history of data points .Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs.

5.1.1. Long Short-Term Memory (LSTM)

In 1997, Hochreiter and Schmidhuber proposed a new network architecture – the LSTM. Recurrent neural networks with Long Short-Term Memory have emerged as an effective and scalable model for several learning problems related to sequential data.
An LSTM contain an Input, output and forget gate. The gates define, if a memory is added, kept or removed from a cell. This way the LSTM learns, which memories should be kept, and which can be “forgotten”.

System Implementation

Language and Tools

The system is implemented in Python 3.5+ and Using the following modules:

- **Tweepy**

  Python Library that is used to fetch tweets from twitter. Twitter Developer account must be provided .

- **Pandas**

  Library for analyzing data from CSV files. We used Pandas for opening and updating CSV files, filtering data, data cleaning and many more.

- **Scikit-learn**

  Great library for data analysis, we have used to in order to implement SVC, preprocessing, feature extraction , cross validation and MinMaxScaler for the LSTM model.

- **NLTK**

  NLTK is a leading platform to work with human language data. We used NLTK in order to implement word tokenizer.

- **Keras**

  Time series analysis based on LSTM can capture the complexity between time steps and predicted value. There are several deep learning methods, TensorFlow, Keras, PyTorch and more. We used Keras since it provides great flexibility and functionality. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with data easier.

- **Matplotlib**

  Plotting library for python, it provides a MATLAB-like interface and used to plot graphs. It also provides easy use with GUI toolkits like Tkinter.

- **Sqlite3**

  Library that is used to create local database that doesn’t require a separate server and allows accessing the database using SQL queries.

- **NumPy**

  NumPy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Methodology

1. Datasets

- **Twitter API**

  We have collected the data using Twitter API. Twitter has limitation for fetching data, therefore, we can only fetch up to 100 tweets per hour for 7 days period. In our project, we are looking up for specific hashtag and collecting all the available tweets that contain the desired hashtag. For each topic, we create a CSV file that contains the labels: Predict (NaN), Date and Time, Tweet.

- **Sentiment140**

  We have used Sentiment140 dataset from Kaggle, the dataset contains 1.6 million tweets that are divided as positive and negative. Sentiment140 will be used as a train/test variable for the sentiment analysis machine learning algorithm.

2. Data Preparation

Social media’s language is usually poor; Therefore, it is very important to maintain those mistakes in order to secure our accuracy. Misspelled words, Links, slangs need to be modified in order to achieve higher accuracy. For each tweet, we have fetched only English letters and then we have preprocessed with the following methods:

2.1. Preprocessing

- Lower case – each word that contain Caps was normalized to lower case (HELLO -> hello)
- URL – each url ‘http://….’ Changed to ‘URL’
- User Mention – replacing ‘@user’ with ‘USER’
- Hashtag – replacing ‘#hashtag’ with an ‘HASHTAG’
- Retweets (RT) – we have removed ‘RT’
- Dots – We have removed the form of multiply dots to space (example : ‘..’ -> ‘ ’)
- Emojis – We have analyzed each emoji and we replace each emoji with POS_EMO/NEG_EMO according to their meaning. (☺ – Positive , ☹ - Negative).
- Remove multiply letters – each word that contains a multiply letter was fixed (example : ‘hellllllo’ - > ‘hello’)
- Remove punctuation (- , ‘ , | , “ , : , ; ….)

2.2. Feature Extraction

2.2.1. Count Vectorizer

Converts the text document collection into a matrix of integers. This method helps to generate a sparse matrix of the counts.

2.2.2. N-grams

For problems like Sentiment Analysis, setting n-gram ranges that use bigrams or trigrams can dramatically improve the accuracy of classification, as they can capture more complex expressions formed by the composition of more than one word. We have split each tweet to a unigram and bigram, taking an N-gram, for example, ‘highly recommended’, will get higher score than just ‘recommended’.

Results over data preprocessing, we have collected data at the finale episode of Games of Thrones and the results:

Tweet : "#GameOfThrones ends tonight - but whatever happens, no-one will be happy

Preprocessed Tweet : hashtag end tonight whatev happens noon happy url

Predicted Value : -1

Tweet: Yet I still call #GameOfThrones my favorite show ever. Because it is. No amount of screwing up this past season can€¦ https://t.co/DxoYDMRVsm

Preprocessed Tweet: yet still call hashtag favorite show ever is amount screw past season url

Predicted Value : 1

3. Sentiment Analysis

We have implemented linear SVM as a classifier so we can classify each tweet with positive or negative label.
We have used Sentiment140 dataset and build a machine which uses K-Stratified Folds, where K=10. Each fold contains different train and test sets which allow us to iterate 10 times over 10 different train and test sets. Each iteration on each fold uses linear Support vector machine algorithm based on CountVectorizer matrix which provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, also to encode new documents using that vocabulary.
Applying Linear SVM algorithm yield accuracy of ~83.3%.

4. Model Preparation

Our data contains a list of labeled tweets and date and time. One of the biggest problems of time-series is small datasets. Since Twitter has limitations on fetching tweets, we must deal with the shape of our time-series and missing data points.

4.1. Time-series shape

We have divided our data by hour, where each hour on a time series represent the ratio between sum positive and sum negative divided by the number of tweets fetched in that hour.

$$Hour\ Score = \frac{[\sum_{n=0}^{n} \text{Positive Tweets} - \sum_{n=0}^{n} \text{Negative Tweets}]}{Tweets\ per\ Hour}$$

4.2. Missing Data

Missing values can be a problem because at each time T, it takes the value of prediction and if there is a missed data point it sets a ‘NaN’ value. Therefore, we had to fill missing values by applying an average between highest data point and lowest data point.

$$Current\ Hour = \frac{[\,Max(Average) + Min(Average)\,]}{2}$$

4.3. Trends and Smoothing

We have used the Moving Average method in order to smooth trends. Given 3 observations, the transformed value at time (t) is calculated as the mean value of the following 3 observations :

$$Observation(t) = \frac{[\,Observation(t-2) + Observation(t-1) + Observation(t)]}{3}$$

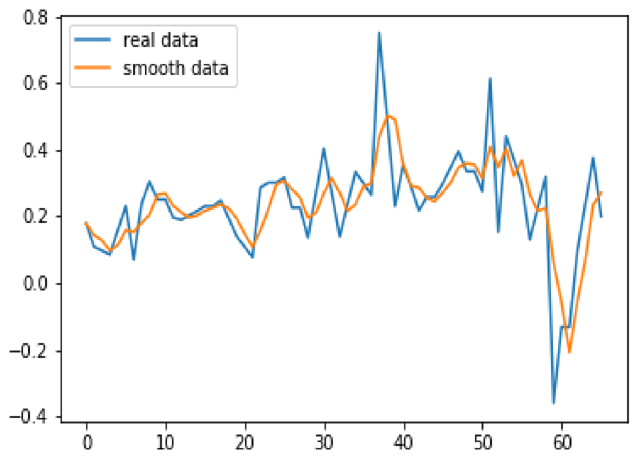


Figure 1: Real and smooth data after performing moving average.

5. Model Derivation and Sentiment Prediction

The purpose of our project is to build a machine that analyze data represented as in time series sequence. The method we used is supervised deep learning method, LSTM which stands for Long-Short term Memory. We will represent our method in this section.

5.1. Data Scaling

We have implemented MinMaxScaler estimator. This estimator scales and translates each feature individually such that it is in the given range on the training set, between -1 and 1. Although our data is between the range of -1 to 1, we have found that using MinMaxScaler actually boost the performance and give better predictions.

5.2. N-steps

One of the main steps is shifting the data into windows, which are small sets of data points of the time series.
Consider the following example of 6-time steps and average:

Date and Time	Average
04/05/2019 21:00	0.178082
04/05/2019 22:00	0.143389
04/05/2019 23:00	0.127893
05/05/2019 00:00	0.096901
05/05/2019 01:00	0.113315
05/05/2019 02:00	0.157938

Predicts :

05/05/2019 03:00	0.153043

We feed our LSTM model with input shapes of 6 Steps each time, in order to predict the next hour in every iteration.

In our experiment, we have notice that the lower the N is, the more stationary data we predict, although, in some cases, higher shape may decrease the performance.

5.3. LSTM Architecture

One hidden layer gives better predictions. More than one layer produces stationary data. We have used 64 Neurons.

```
n_steps=6

n_features=1

model.add (LSTM (units=64, activation='tanh', input_shape= (n_steps, n_features), bias_initializer='ones')
```

We implemented 1-layer network with 64 neurons, Input shape has 6 inputs and 1 output. The output layer receives connections only from memory cells. Memory cells and gate units receive inputs from input units, memory cells and gate units, and have bias weights.

5.3.1. Optimizer and Loss

Since we are working on a regression, we implemented the Mean Squared Error (MSE) loss function. Mean squared error is calculated as the average of the squared differences between the predicted and actual values. The result is always positive regardless of the sign of the predicted and actual values , a perfect value is 0.0. The squaring means that larger mistakes result in more error than smaller mistakes, meaning that the model is punished for making larger mistakes.

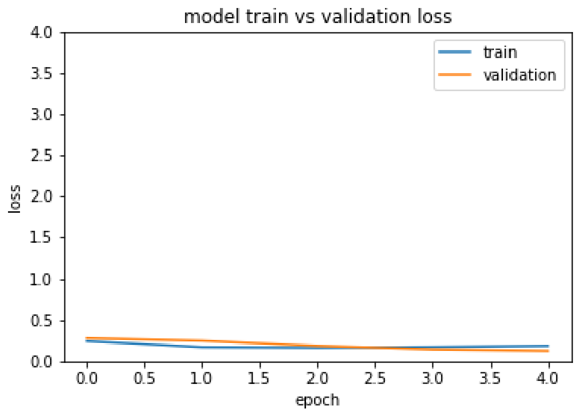


Figure 2: Training loss vs Validation Loss

5.3.2. Epoch and Early Stopping

Both learning rate decay and early stopping are well-known, easy regularization techniques that prevents overfitting. The idea behind a decaying learning rate is to start big, in order to punish big mistakes when finding the gradient and reducing the learning rate over time to 0. Early stopping is a logical condition that simply stops training when the testing error stops improving for a fixed number of iterations. Which eventually results in the best fit.

Results

The results of the two models are presented below. Increasing the amount of data, in quantity consistently improves both models, which would indicate that an accurate model is feasible if relevant data of enough quantity is given.

Experiment 1

Dataset: #Trump

Duration: 5 Days

% For prediction : 20% (about 24 hours prediction)

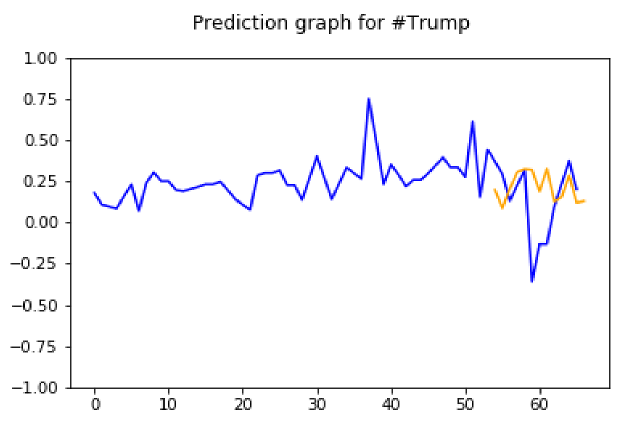


Figure 3 : Prediction result for #Trump - Non-Stationary Data

```
Train on 48 samples, validate on 12 sample - Non-Stationary Data

Epoch 1/100 - 1s 13ms/step - loss: 0.4456 - val_loss: 0.2246

Epoch 2/100 - 0s 325us/step - loss: 0.2024 - val_loss: 0.0669

Epoch 3/100  - 0s 325us/step - loss: 0.1480 - val_loss: 0.0591

Epoch 4/100  - 0s 0us/step - loss: 0.1770 - val_loss: 0.0640

Epoch 5/100 - 0s 326us/step - loss: 0.752 - val_loss: 0.0559

Epoch 00005: early stopping
```

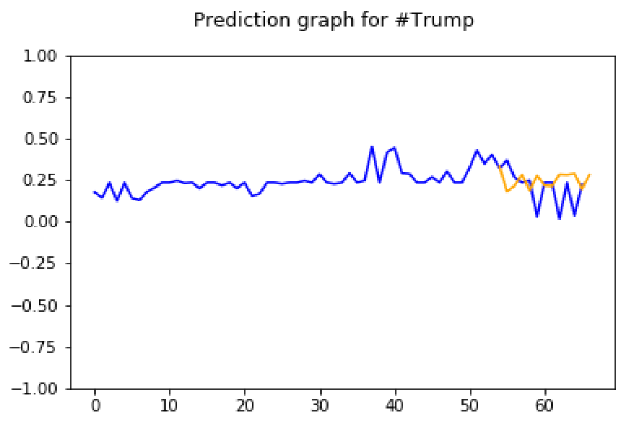


Figure 4 : Prediction result for #Trump – Stationary Data

```
Train on 48 samples, validate on 12 samples – Stationary Data

Epoch 1/100 - s 23ms/step - loss: 2.6747 - val_loss: 1.8638

Epoch 2/100 - s 312us/step - loss: 1.7021 - val_loss: 1.0669

Epoch 3/100 - s 312us/step - loss: 0.9906 - val_loss: 0.5233

Epoch 4/100 - s 332us/step - loss: 0.5266 - val_loss: 0.2099

Epoch 5/100 - s 353us/step - loss: 0.2824 - val_loss: 0.0825

Epoch 6/100 - s 291us/step - loss: 0.2076 - val_loss: 0.0754

Epoch 7/100 - s 249us/step - loss: 0.2312 - val_loss: 0.1139

Epoch 8/100 - s 353us/step - loss: 0.2794 - val_loss: 0.1427

Epoch 00008: early stopping
```

Experiment 2
Dataset: #Trump
Duration: 5 Days
% For prediction : 5% (about 6 hours prediction)

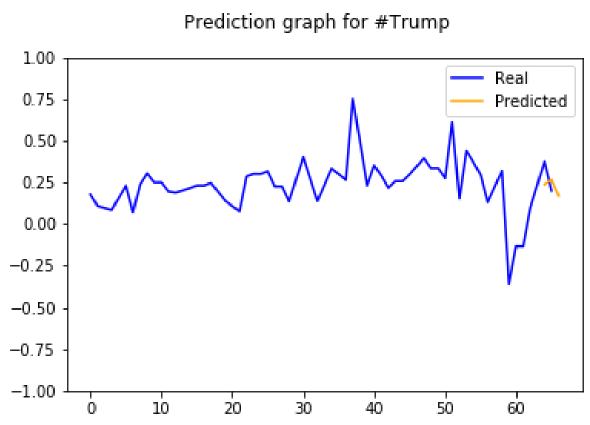


Figure 5: Prediction result for #Trump – Non-Stationary data

```
Train on 48 samples, validate on 12 sample - Non-Stationary Data

Epoch 1/100 - 1s 14ms/step - loss: 1.2675 - val_loss: 1.2732

Epoch 2/100 - 0s 208us/step - loss: 0.6226 - val_loss: 0.6720

Epoch 3/100 - 0s 249us/step - loss: 0.2321 - val_loss: 0.3191

Epoch 4/100 - 0s 208us/step - loss: 0.0721 - val_loss: 0.1730

Epoch 5/100 - 0s 208us/step - loss: 0.0780 - val_loss: 0.1504

Epoch 6/100 - 0s 229us/step - loss: 0.1439 - val_loss: 0.1590

Epoch 7/100 - 0s 249us/step - loss: 0.1790 - val_loss: 0.1570

Epoch 8/100 - 0s 249us/step - loss: 0.1620 - val_loss: 0.1498
```

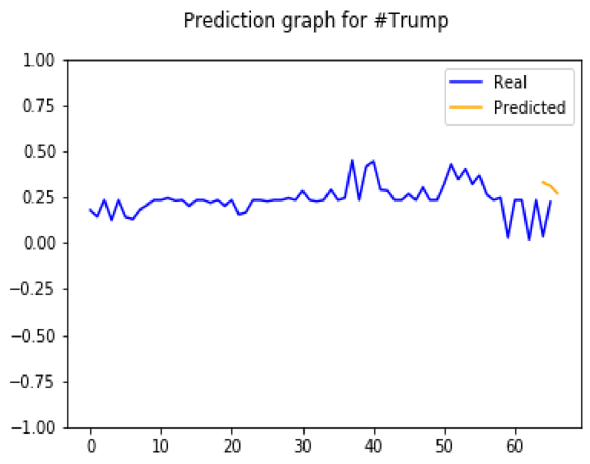


Figure 6 : Prediction result for #Trump – Stationary Data

```
Train on 48 samples, validate on 12 samples – Stationary Data

Epoch 1/100 - 1s 13ms/step - loss: 0.3556 - val_loss: 0.3146

Epoch 2/100 - 0s 187us/step - loss: 0.1486 - val_loss: 0.2336

Epoch 3/100 - 0s 166us/step - loss: 0.1226 - val_loss: 0.2342

Epoch 4/100 - 0s 166us/step - loss: 0.1309 - val_loss: 0.2324

Epoch 5/100 - 0s 187us/step - loss: 0.1163 - val_loss: 0.2297

Epoch 6/100 - 0s 166us/step - loss: 0.0966 - val_loss: 0.2454

Epoch 7/100 - 0s 187us/step - loss: 0.0957 - val_loss: 0.2758

Epoch 8/100 - 0s 166us/step - loss: 0.1118 - val_loss: 0.3002
```

Conclusion and Future Work

In this project, we have shown that our proposed approach of predicting score of a sentiment in time series using neural networks based on sentiment analysis is possible and our experiment shows that both stationary and Non-stationary data performs well, but since we are trying to predict independent data points, it is almost impossible to predict extreme data points with 0 loss.
Therefore, future work should consider using the amount of data in each hour in order to scale the real values and ignore cases that only few tweets determine the value of specific hour . by that, it is possible to reduce unexpected extreme values.
Stationary data performs better on statistical methods rather than regression. It might be recommended to use linear interpolation to handle trends.

While our results are very promising, and increases with the data, future work should follow up on decreasing the loss even more and achieve higher precision in prediction.
We would also recommend increasing the performance of the sentiment analysis in order to increase the accuracy.

References

1. B. Wang, M. Liu (2015). Deep Learning for Aspect-Based Sentiment Analysis.

2. Mehmet Kaya and Shannon Conley (2016). Comparison of sentiment lexicon development techniques for event prediction

3. Nikhil Kumar Singh, Deepak Singh Tomar, Arun Kumar Sangaiah (2018). Sentimental Analysis : A review and comparative analysis over social media.

4. Sayan Mukhopadhyay , Kolkata, West Bengal, India (2018). *Advanced Data Analytics Using Python*

5. Alec Go, Lei Huang , Richa Bhayani (2009). Twitter Semantic analysis, Stanford university.

6. Mina Jeon, Sanghoon Jun, and Eenjun Hwang (2014). Web-age information Management , Hashtag Recommendation Based on User Tweet and Hashtag Classification on Twitter.

7. Ali Alessa and Miad Faezipour (2018). Machine Learning and data Mining in Pattern Recognition . School of Engineering, University of Bridgeport, Bridgeport, CT

8. Anthony K Jose , Nipun Bhatia Sarath Krishna S. (2010). Twitter Sentiment Analysis

9. Marta Arias, Argimiro Arratia, Ramon Xuriguera (2012). Forecasting with Twitter Data. Universitat Politecnica de Catalunya.

10. Assoc Prof. Dr. Sevtap Kestel (2013). Time Series Analysis

11. Robert, K.L., Chin-Yuan, F., Wei-Hsiu, H., Pei-Chan, C. (2009). Evolving and clustering fuzzy decision tree for financial time series data forecasting. Expert Systems with Appli. 4, 3761–3773

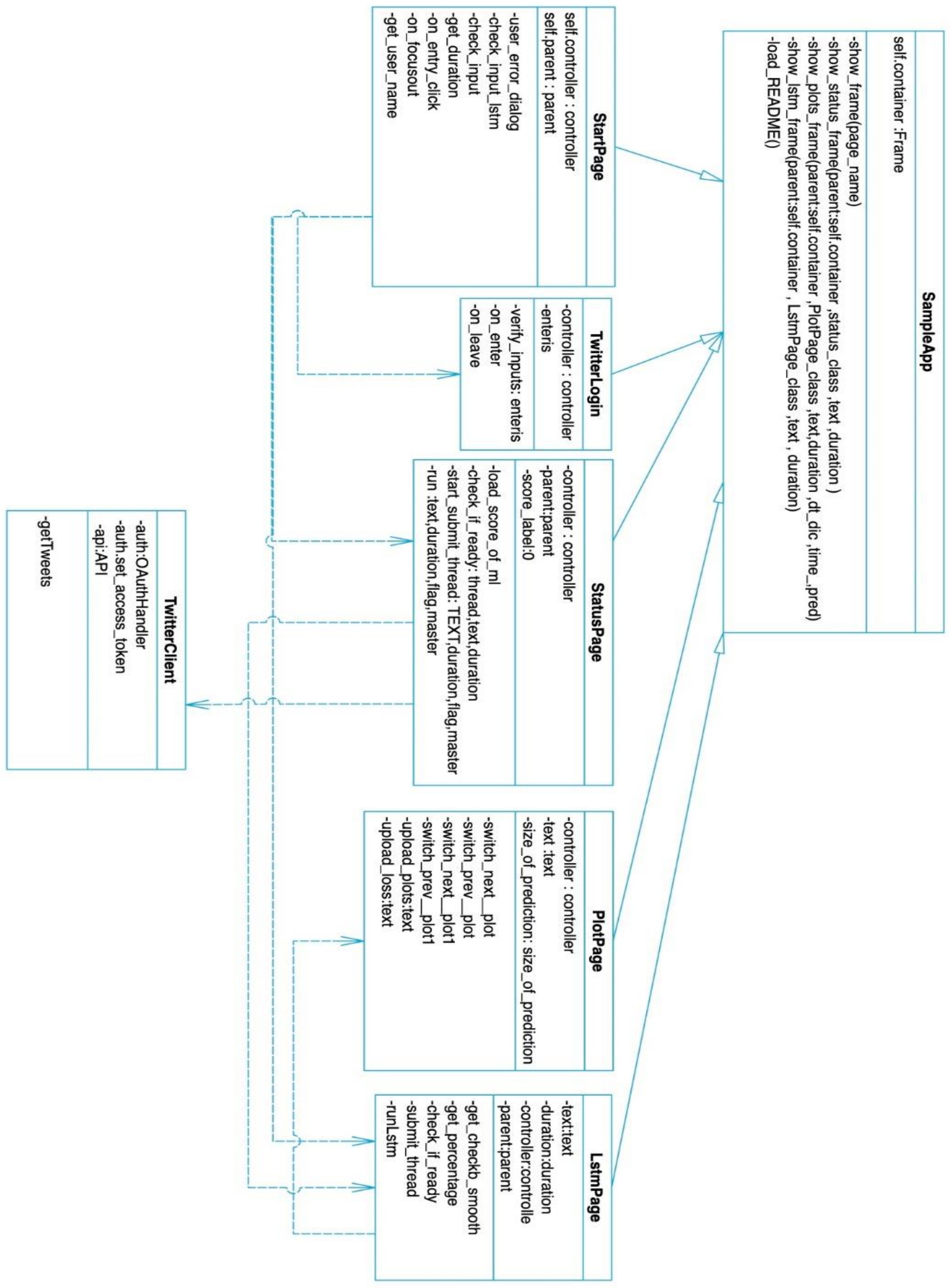
12. Tamanna Hasib, Saima Ahmed Rahin . spect-Based Sentiment Analysis Using SemEval and Amazon Datasets, BRAC University.

13. Edwin Li (2018). LSTM Neural Network Models for Market Movement Prediction.

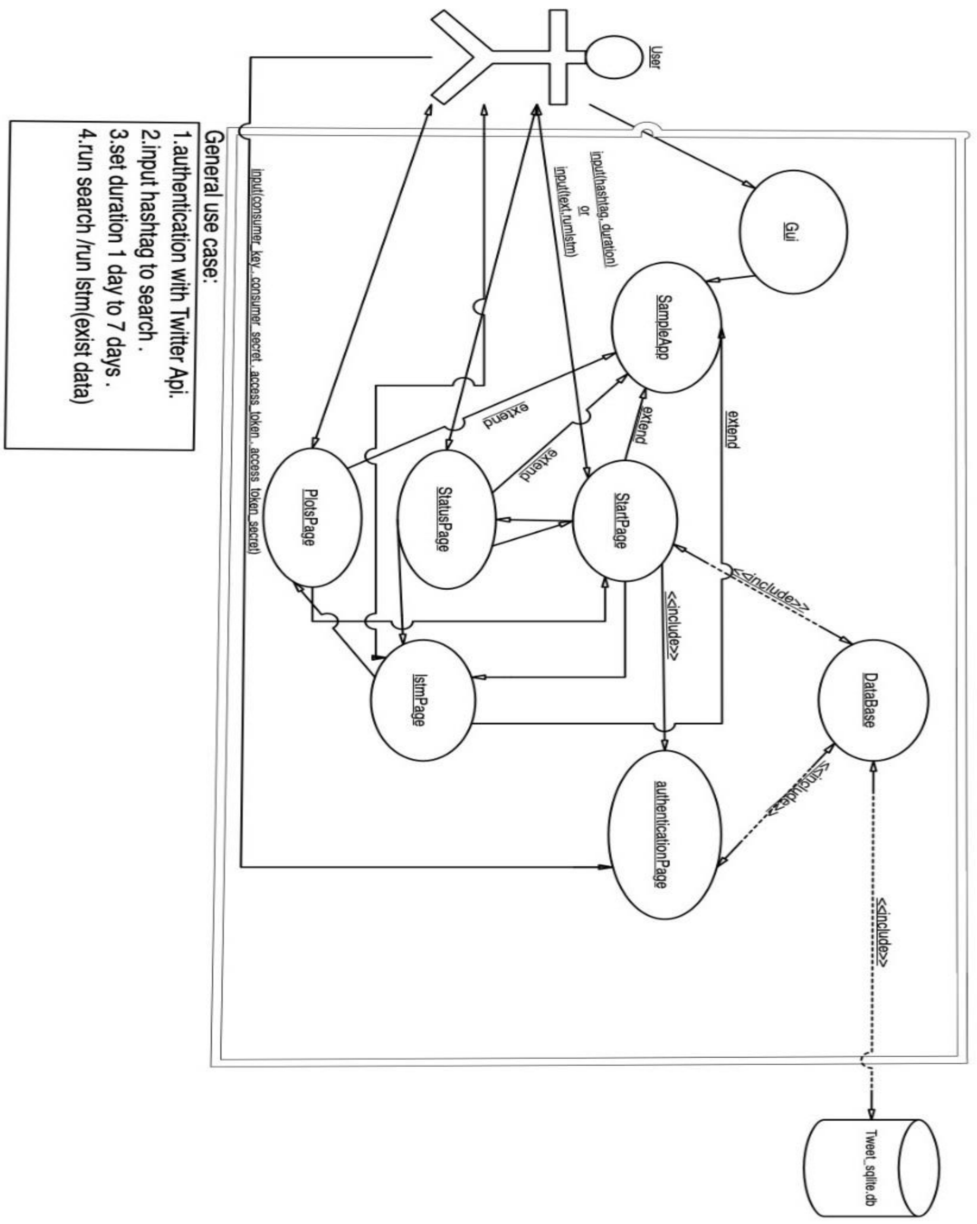
14. S. Hochreiter and J. Schmidhuber (1997). “Long short-term memory”.

15. Anshul Mittal and Arpit Goel (2011) .Stock Prediction Using Twitter Sentiment Analysis . Stanford University

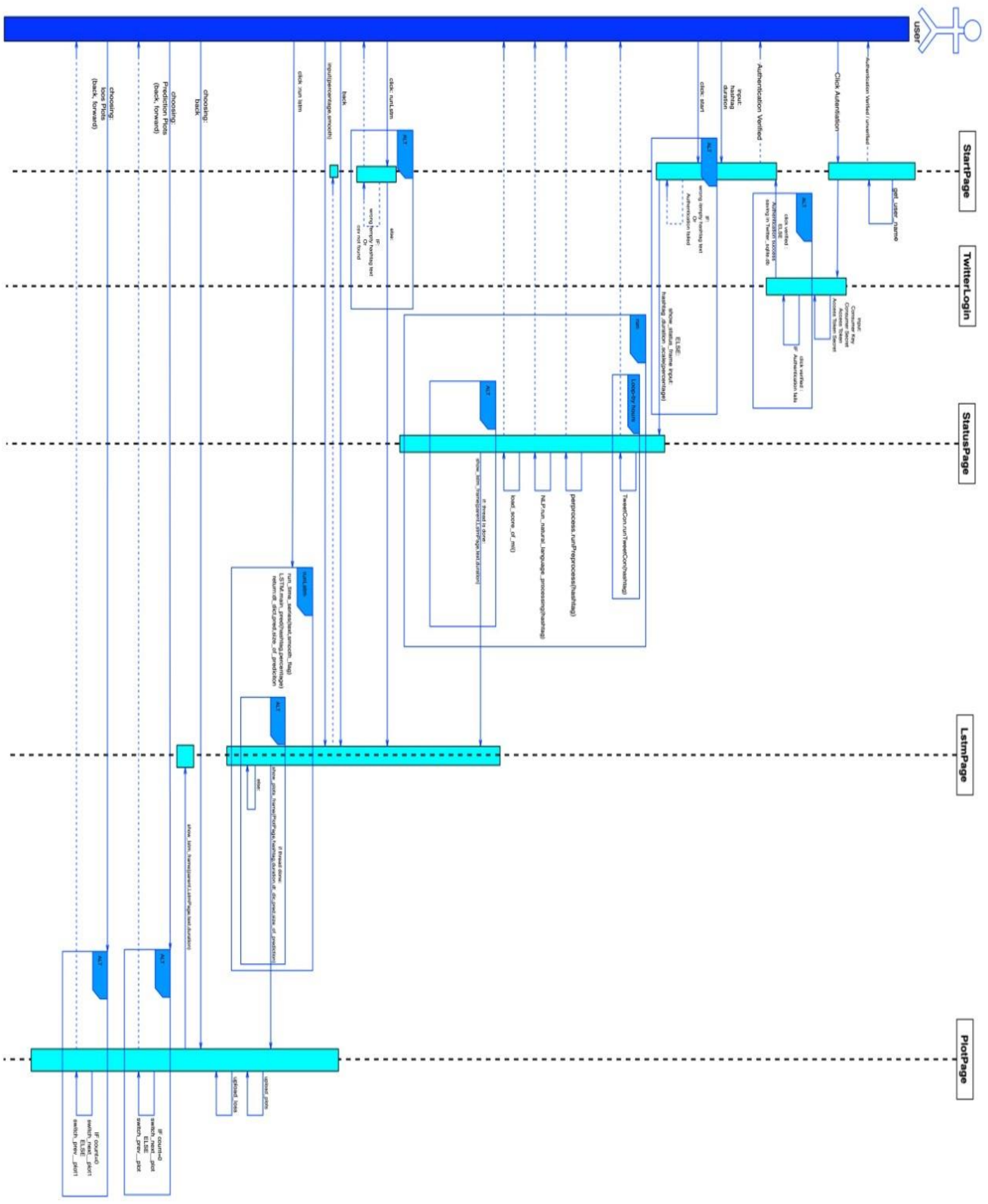
Class Diagram

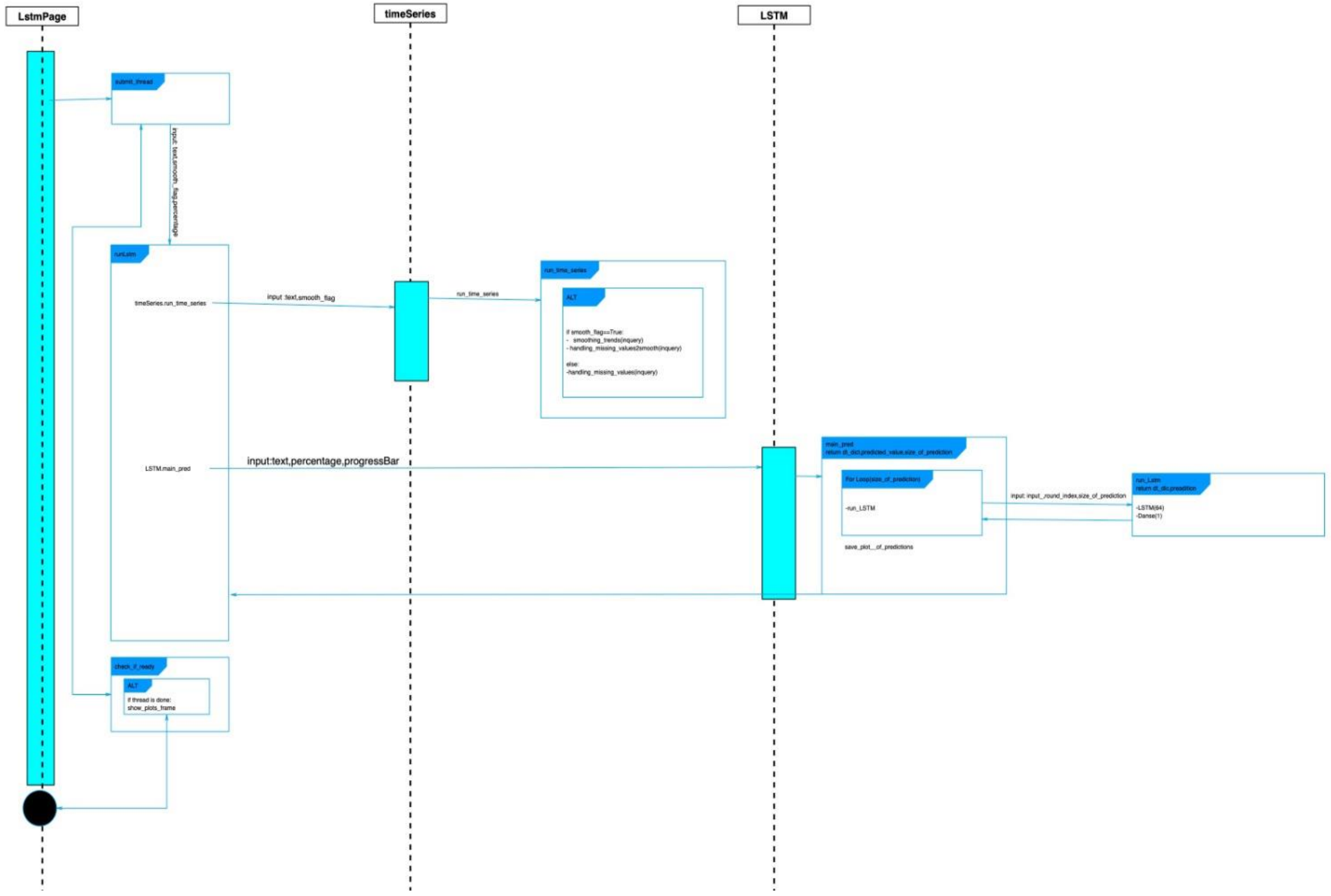


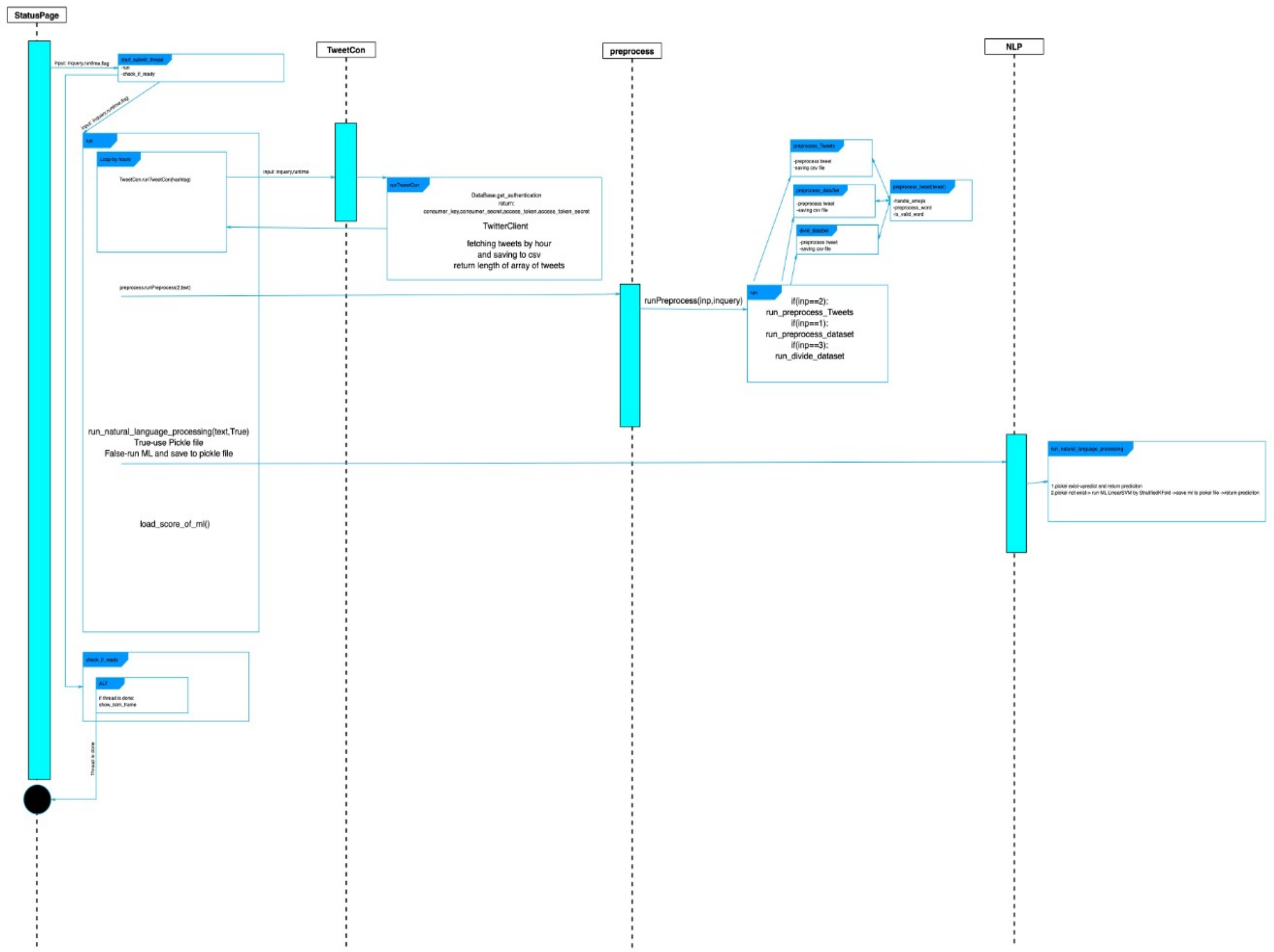
Use Case Diagram



Sequence Diagrams




LstmPage
timeSeries
LSTM
submit_thread
input: text,smooth_flag,percentage
runLstm
timeSeries.run_time_series
input :text,smooth_flag
run_time_series
run_time_series
ALT
if smooth_flag==True:
- smoothing_trends(inquery)
- handling_missing_values2smooth(inquery)
else:
-handling_missing_values(inquery)
LSTM.main_pred
input:text,percentage,progressBar
main_pred
return dt_dict,predicted_value,size_of_prediction
For Loop(size_of_prediction)
-run_LSTM
save_plot__of_predictions
input: input_,round_index,size_of_prediction
run_Lstm
return dt_dic,preadition
-LSTM(64)
-Danse(1)
check_if_ready
ALT
if thread is done:
show_plots_frame


StatusPage
TweetCon
preprocess
NLP
input: inquery,runtime,flag
start_submit_thread
-run
-check_if_ready
input: inquery,runtime,flag
run
Loop-by hours
TweetCon.runTweetCon(hashtag)
input: inquery,runtime
runTweetCon
DataBase.get_authentication
return:
consumer_key,consumer_secret,access_token,access_token_secret
TwitterClient
fetching tweets by hour
and saving to csv
return length of array of tweets
preprocess.runPreprocess(2,text)
runPreprocess(inp,inquery)
run
if(inp==2):
run_preprocess_Tweets
if(inp==1):
run_preprocess_dataset
if(inp==3):
run_divide_dataset
preprocess_Tweets
-preprocess tweet
-saving csv file
preprocess_dataSet
-preprocess tweet
-saving csv file
divid_dataSet
-preprocess tweet
-saving csv file
preprocess_tweet(tweet)
-handle_emojis
-preprocess_word
-is_valid_word
run_natural_language_processing(text,True)
True-use Pickle file
False-run ML and save to pickle file
run_natural_language_processing
1.pickel exist->predict and return prediction
2.pickel not exist-> run ML LinearSVM by StratifiedKFold ->save ml to pickel file ->return prediction
load_score_of_ml()
check_if_ready
ALT
if thread is done:
show_tsm_frame
Thread is done

Sentiment time-series Prediction In Tweets with LSTM

מערכת לחיזוי רגשות בציוצים על גבי ציר זמן בשימוש ב-LSTM

נכתב על ידי : בר גבאי וירון מרדכי

מנחה אקדמי : ד"ר מרינה ליטבק

יעדים

כללי – הבהקים

- מערכת אשר חוזה רגש עתידי לפי רגשות משתמשי טוויטר על גבי ציר-זמן.
- ניתוח רגשות בעל דיוק גבוה, עבור כל האשטאג שהמשתמש ביקש.
- המטרה העיקרית של מערכת זו, היא להציג ללקוח את החיזוי האפשרי של אותו האשטאג.
- המערכת מתבססת על בסיס נתונים שאותו שואבים דרך API של הטוויטר.
- במערכת זו יש משתמש אחד – הלקוח.

בעיות

בעיות נוכחיות

ביום קיימות מערכות דומות, אך כל מערכת שקיימת פועלת בדרכים שונות. יש מערכות בעלות דיוק גבוהה, ויש כאלו עם דיוק נמוך יותר. בפרויקט זה נשאף להשיג את הדיוק הגבוהה ביותר עבור חיזוי האירועים.

ישימות ועלות/תועלת

סיכונים - ישימות הפרויקט

מעט מידע – מעט מידע יכול לגרום לנקודות קיצוניות בגרף, ולכן נרצה להפוך את הנקודות על הגרף להיות יציבות .

מאפיינים כלליים

מצב קיים

ביום קיימות מערכות שונות לחזות אירועים, אך טרם נתחו נתונים בזמן אמת בטוויטר.

אופי המערכת וסוגה

המערכת תשאף להתעלות בדיוק החיזוי ביחס למערכות אחרות. אך לא להחליף מערכות אחרות, שכן , כל אחת מהן פועלת בדרך אחרת.

אילוצים

המערכת תכתב בשפת תכנות Python ולכן יכולה לרוץ על כל מערכת הפעלה.

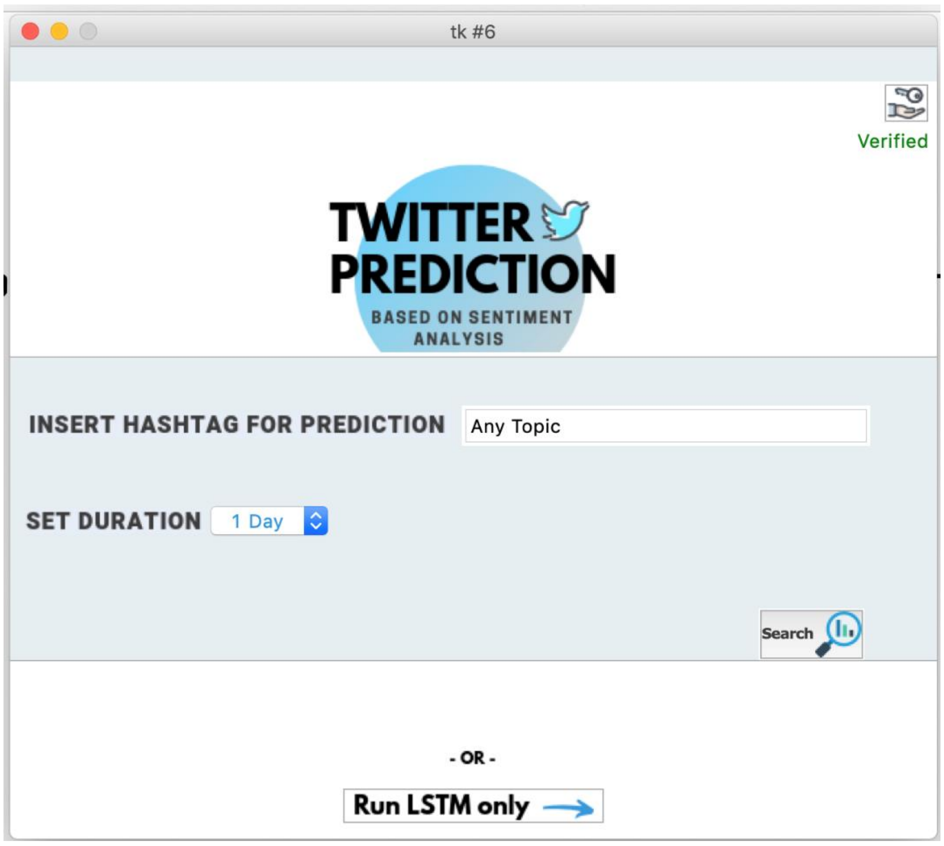
תיחום חיצוני

תיחום כללי

כניסה למערכת תהיה נגישה עבור כל משתמש שבבעלותו משתמש developer של טוויטר.

<u>ממשק משתמש</u>

מסך ראשי



INSERT HASHTAG FOR PREDICTION: Any Topic

SET DURATION: 1 Day

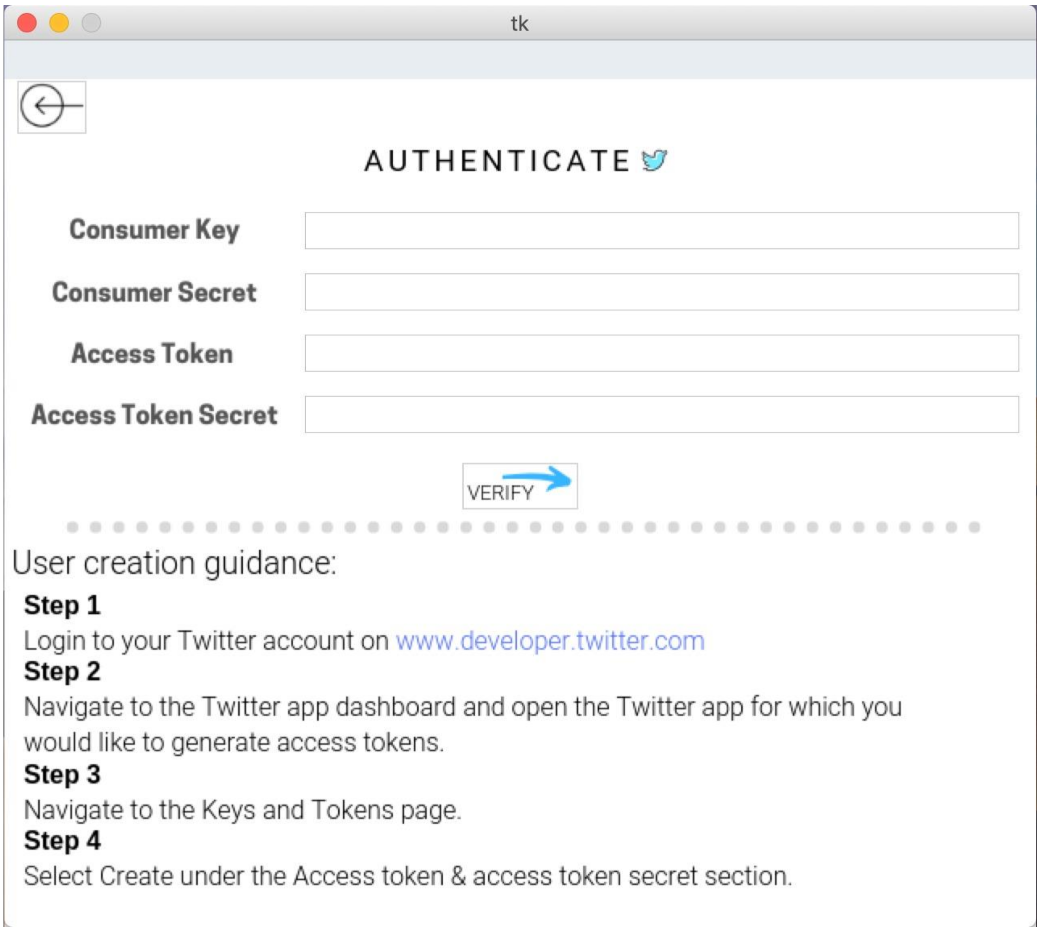
<u>3 אפשרויות:</u>

1.הכנסת אשטאג ובחירת זמן ריצה לפי ימים- עד 7 ימים ולאחר מכן חיפוש.

2.הכנסת אשטאג ולחיצה run lstm (במקרה שהאשטג קיים בנתונים).

3.התחברות לtwitter api

מסך אימות – טוויטר

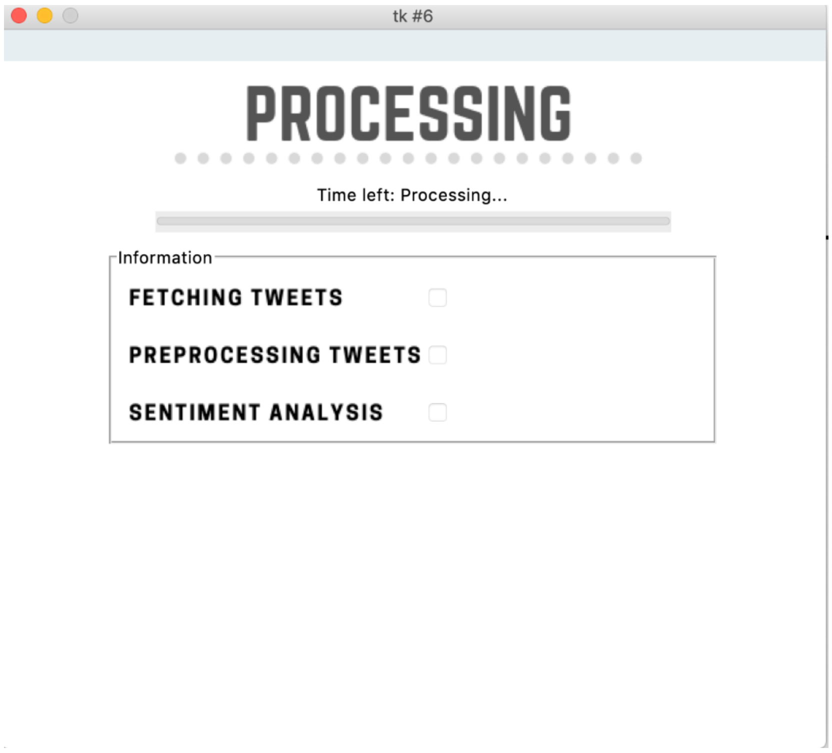


User creation guidance:

Step 1
Login to your Twitter account on www.developer.twitter.com
Step 2
Navigate to the Twitter app dashboard and open the Twitter app for which you would like to generate access tokens.
Step 3
Navigate to the Keys and Tokens page.
Step 4
Select Create under the Access token & access token secret section.

הזנת פרטים אישיים של חשבון Twitter Developer, בלחיצה על כפתור "Verify" תתבצע אימות עם שרת Twitter .

מסך Processing



Time left: Processing...

Information

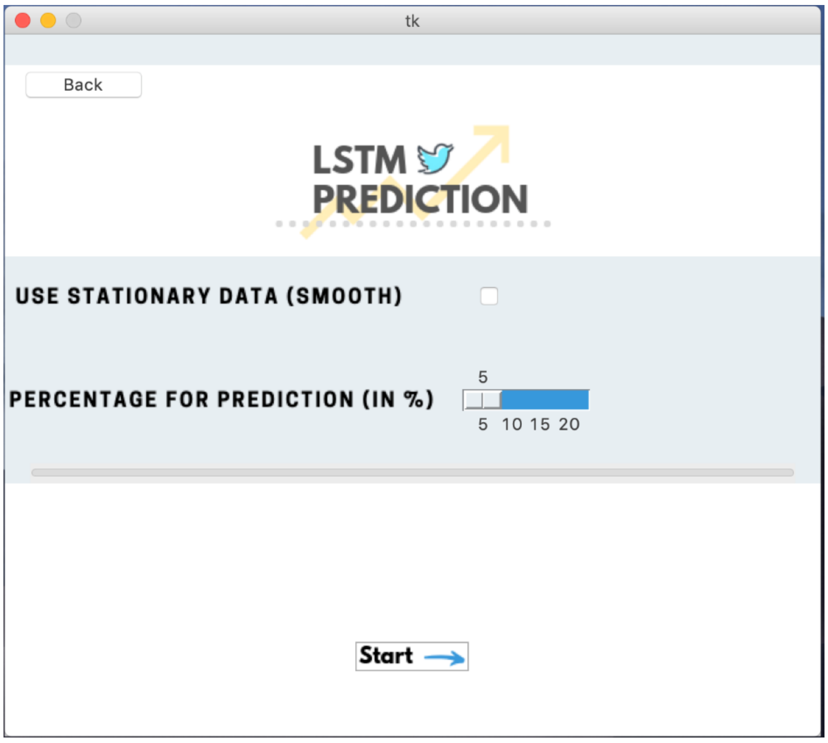
FETCHING TWEETS ☐

PREPROCESSING TWEETS ☐

SENTIMENT ANALYSIS ☐

במסך זה מתבצעות פעולות ברקע בלבד

מסך LSTM PREDICTION


tk

Back

LSTM PREDICTION

USE STATIONARY DATA (SMOOTH)

PERCENTAGE FOR PREDICTION (IN %) 5

5 10 15 20

Start

4 אפשרויות:

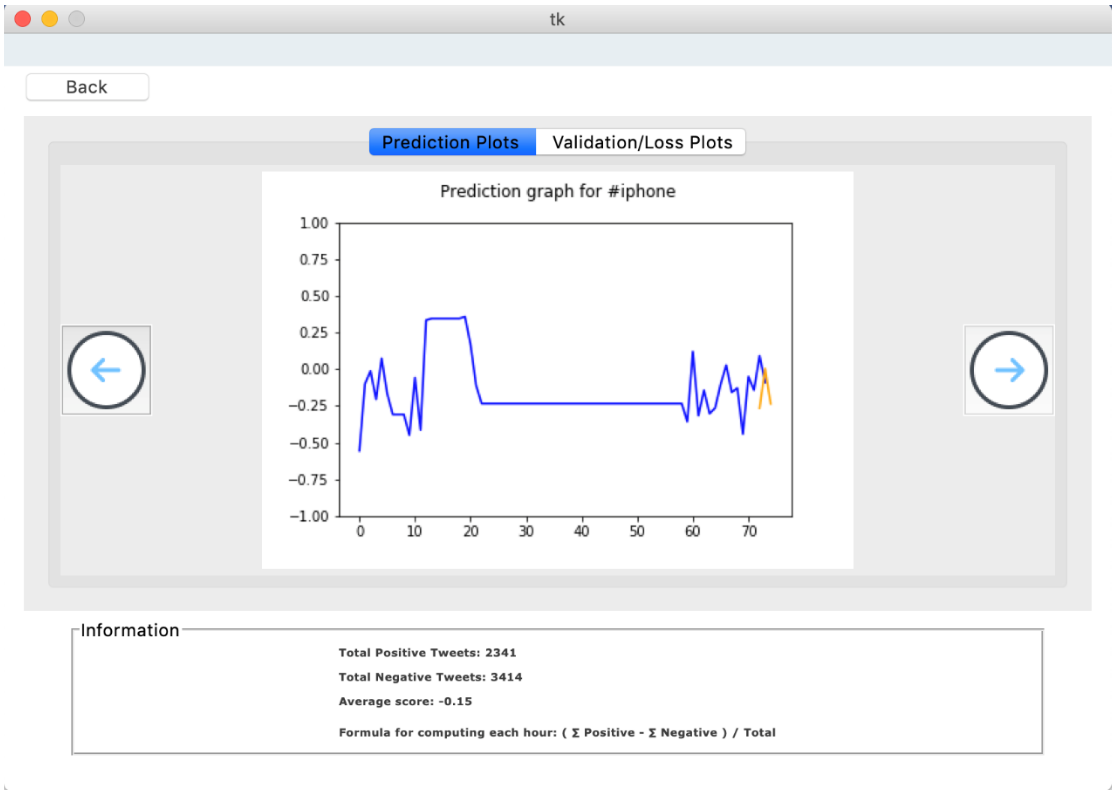
1.בחירת smooth data.

2.בחירת אחוזים לחיזוי עד 20% מכמות השעות שנשאבו.

3.start lstm

4.חזור למסך הראשי.

מסך Plots



Total Positive Tweets: 2341

Total Negative Tweets: 3414

Average score: -0.15

Formula for computing each hour: (Σ Positive - Σ Negative) / Total

<u>3 אפשרויות:</u>

1.בחירת דף Validation/loos Plots / prediciton plots.

2.בחירת תמונות ימינה/שמאלה.

3.חזרה לדף הlstm.

4.0 כללי – הבהקים

המימוש יתנהל לפי

גורמים מעורבים – ניהול, צוות פיתוח וסיוע טכני

1) תפקיד – מנהל, מפתח

שם – ירון מרדכי

Email- yaronMord@gmail.com

2) תפקיד – מפתחת

שם – בר גבאי

Email- bargabay92@gmail.com

4.2 תכנית עבודה

4.2.0 שיטת הפיתוח

נשתמש בשיטת הפיתוח של AGILE. תוך כדי התייחסות לכל השלבי עבודה בשיטה זו.