

SENTIMENT ANALYSIS ON UKRAINE RUSSIA'S WAR TWEETS

A MINI PROJECT REPORT

SUBMITTED BY

LOGESHWARAN A	113219031078
JANAKIRAMAN R	113219031056
ANBARASU A	113219031012
HARISH M	113219031050

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING**



AN AUTONOMOUS INSTITUTION

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2022

BONAFIDE CERTIFICATE

Certified that this Mini project report “**SENTIMENT ANALYSIS ON UKRAINE RUSSIA’S TWEETS**” is the Bonafede work of **HARISH M (113219031050), JANAKIRAMAN R (113219031056) , ANBARASU A (113219031013) , LOGESHWARAN A (113219031078)** who carried out the Mini project work under my supervision.

SIGNATURE

MRS. C. BHARATHI SRI

SUPERVISOR

Computer Science and Engineering
Velammal Engineering College
Ambattur – Redhill’s Road,
Chennai – 600066.

SIGNATURE

DR.B. MURUGESWARI

HEAD OF THE DEPARTMENT

Computer Science and Engineering,
Velammal Engineering College,
Ambattur–Redhill’s Road,
Chennai – 600066.

MINI PROJECT EXAMINATION

The Mini project work of this project “**SENTIMENT ANALYSIS ON UKRAINE RUSSIA’S WAR TWEETS**” is a Bonafede record of project done at Department of Computer Science and Engineering, Velammal Engineering College during the academic year 2021 - 2022 by

LOGESHWARAN A	113219031078
JANAKIRAMAN R	113219031056
ANBARASU A	113219031012
HARISH M	113219031050

Of third year of Engineering in Computer Science and Engineering Submitted for university examination held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

The Russo-Ukrainian War is an ongoing war which began in February 2014 between Russia (together with pro-Russian separatist forces) and Ukraine. Both nations are former members of the now dissolved communist Soviet Union prior to its dissolution in 1991. Various agreements were made when they separated in 1991 and each started transitioning towards market-oriented economies. By 2013, much of Ukraine and its national parliament wanted closer ties with Western Europe. On the other hand, others including the then-President Viktor Yanukovych and his cabinet, wanted closer ties with Russia. Before the conflict was resolved though, a series of large-scale, mostly peaceful protests began. Soon, violence broke out, especially in January and February 2014, resulting in the overthrow of the pro-Russian Yanukovych. As a result of this, the parliament appointed an interim government seeking greater ties with Western Europe. Soon Russia initiated military action, initially focused on the status of Crimea and the Donbas, internationally recognized as part of Ukraine. The first eight years of the conflict included the Russian annexation of Crimea (2014) and the war in Donbas (2014–present) between Ukraine and Russian-backed separatists, as well as naval incidents, cyberwarfare, and political tensions. Following a Russian military build-up on the. On 21 February 2022, Russia officially recognized the two self-proclaimed separatist states in the Donbas, and openly sent troops into the territories. Three days later, Russia invaded Ukraine. Much of the international community has condemned Russia for its actions in post-revolutionary Ukraine, accusing it of breaking international law and violating Ukrainian sovereignty. Many countries implemented economic sanctions against Russia, Russian individuals, or companies, especially after the 2022 invasion. In this project we are going to analyze the sentiment of people's tweets and gain insights from it.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	4
	LIST OF FIGURES	7
	LIST OF ABBREVIATIONS	8
1	INTRODUCTION	
	1.1 PURPOSE OF THE PROJECT	9
	1.2 LAYERED ARCHITECTURE	9
2	SYSTEM SPECIFICATION	
	2.1 HARDWARE REQUIRENMENT	11
	2.2 SOFTWARE REQUIRENMENT	11
	2.3 TECHNOLOGIES USED	11
3	PROBLEMS IN SUPERVISED LEARNING	
	3.1 HETEROGENEITY OF DATA	12
	3.2 REDUNDANCY OF DATA	12
	3.3 DEPENDENT FEATURES	12
	3.4 BIAS-VARIANCE TRADEOFF	12
	3.5 CURSE OF DIMENSIONALITY	13
	3.6 OVERFITTING	13

4	PROPOSED SOLUTIONS	14
5	DATASET	
	5.1 TWITTER API	15
	5.2 GATHERING TWEETS	15
	5.3 IMPLEMENTATION	16
	5.4 FEATURES	19
6	ANALYSIS	
	6.1 FRAMEWORKS USED	20
	6.2 INSIGHTS	20
7	TRAINING A NEURAL NETWORK	
	7.1 TRAIINT_TEST_SPLIT	27
	7.2 BERT MODEL	28
	7.3 LSTM	34
	7.4 PREDICTIONS	40
8	CONCLUSION & REFERENCES	44

LIST OF FIGURES

FIG.NO	NAME OF THE FIGURE	PAGE NO
1.1	Layered Architecture of ML model	9
5.1	Gathering data from Twitter API	16
5.3	Data Frame	19
6.2.1	Reviewed Data Frame	25
6.2.2	Top 10 User Location	29
6.2.3	Distribution of sentiments	31
6.2.4	Sentiment Polarity	37
7.4	Model Layers	38

LIST OF ABBREVIATIONS

TERMS	ABBREVIATION
SQL	Structured Query Language
BERT	Bidirectional Encoder Representations from Transformers
NLP	Natural Language Processing
RNN	Recurrent Neural Network
LSTM	Long Term Short Memory

CHAPTER 1

INTRODUCTION

1.1 Purpose of the Project

The Russo-Ukrainian War is an ongoing war which began in February 2014 between Russia (together with pro-Russian separatist forces) and Ukraine.

- The purpose of the project is to identify people's opinion about this ongoing war.
- To identify how people in different countries are reacting to war.
- To analyze sentiment of the tweet dataset.
- To predict sentiment based on given input.

1.2 Layered Architecture

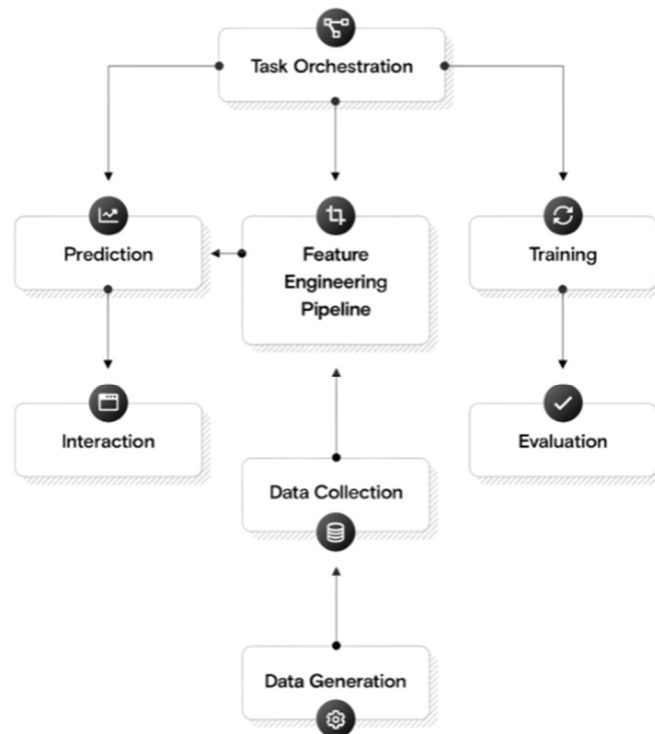


Fig 1.1 layered architecture

- **Data Generation:** Every machine learning application lives off data. That data has to come from somewhere. Usually it's generated by one of your core business functions. In our case it's from twitter api.
- **Data Collection:** Data is only useful if it's accessible, so it needs to be stored – ideally in a consistent structure and conveniently in one place.
- **Feature Engineering Pipeline:** Algorithms can't make sense of raw data. We have to select, transform, combine, and otherwise prepare our data so the algorithm can find useful patterns.
- **Training:** This is where the magic happens. We apply algorithms, and they learn patterns from the data. Then they use these patterns to perform tasks.
- **Evaluation:** We need to carefully test how well our algorithm performs on data it hasn't seen before (during training). This ensures we don't use prediction models that work well on "seen" data, but not in real-world settings.
- **Task Orchestration:** Feature engineering, training, and prediction all need to be scheduled on our computer infrastructure (such as AWS or Azure) – usually with non-trivial interdependence. So we need to reliably orchestrate our tasks.
- **Prediction:** This is the moneymaker. We use the model we've trained to perform new tasks and solve new problems – which usually means making a prediction.
- **Infrastructure:** Even in the age of the cloud, the solution has to live and be served somewhere. This will require setup and maintenance.
- **Authentication:** This keeps our models secure and makes sure only those who have permission can use them.
- **Interaction:** We need some way to interact with our model and give it problems to solve. Usually this takes the form of an API, a user interface, or a command-line interface.
- **Monitoring:** We need to regularly check our model's performance. This usually involves periodically generating a report or showing performance history in a dashboard.

CHAPTER 2

SYSTEM SPECIFICATION

3.1 Hardware Requirement

- CPU: Intel core I3 – 4130 CPU @ 3.40GHz.
- OPERATING SYSTEM: Windows 7 or above
- RAM: 4GB.
- HARD DISK: 32GB

3.2 Software Requirement

- MySQL Community Server
- Google colab or Jupiter notebook
- Any Browser with latest updates
- Any spreadsheet software
- Tableau

3.3 Technologies Used

- Python for Deep Learning
- Sql for basic Data Retrieving

CHAPTER 3

PROBLEMS IN SUPERVISED LEARNING

To intelligently pick an algorithm to use for a supervised learning task, we must consider the following factors:

3.1. Heterogeneity of Data:

Many algorithms like neural networks and support vector machines like their feature vectors to be homogeneous numeric and normalized. The algorithms that employ distance metrics are very sensitive to this, and hence if the data is heterogeneous, these methods should be the afterthought. Decision Trees can handle heterogeneous data very easily.

3.2. Redundancy of Data:

If the data contains redundant information, i.e. contain highly correlated values, then it's useless to use distance based methods because of numerical instability. In this case, some sort of Regularization can be employed to the data to prevent this Situation.

3.3. Dependent Features:

If there is some dependence between the feature vectors, then algorithms that monitor complex interactions like Neural Networks and Decision Trees fare better than other algorithms.

3.4. Bias-Variance Tradeoff:

A learning algorithm is biased for a particular input x if, when trained on each of these data sets, it is systematically incorrect when predicting the correct output for x , whereas a learning algorithm has high variance for a particular input x if it predicts different output values when trained on different training sets. The prediction error of a learned classifier can be related to the sum of bias and variance of the learning algorithm, and neither can be high as they will make the prediction error to be high. A key feature of machine learning algorithms is that they are able to tune the balance between bias and variance automatically, or by manual tuning using bias parameters, and using such algorithms will resolve this situation.

3.5. Curse of Dimensionality:

If the problem has an input space that has a large number of dimensions, and the problem only depends on a subspace of the input space with small dimensions, the machine learning algorithm can be confused by the huge number of dimensions and hence the variance of the algorithm can be high. In practice, if the data scientist can manually remove irrelevant features from the input data, this is likely to improve the accuracy of the learned function. In addition, there are many algorithms for feature selection that seek to identify the relevant features and discard the irrelevant ones, for instance Principal Component Analysis for unsupervised learning. This reduces the dimension.

3.6. Overfitting:

The programmer should know that there is a possibility that the output values may constitute an inherent noise which is the result of human or sensor errors. In this case, the algorithm must not attempt to infer the function that exactly matches all the data. Only after considering all these factors can we pick a supervised learning algorithm that works for the dataset we are working on. For example, if we were working with a dataset consisting of heterogeneous data, then decision trees would fare better than other algorithms. If the input space of the dataset we were working on had 1000 dimensions, then it's better to first perform PCA on the data before using a supervised learning algorithm on it.

CHAPTER 4

PROPOSED SOLUTIONS

- The heterogeneity problem in our dataset will rarely occur as we are gathering data from twitter api. It will be automatically scalable and does not contain outliers or missing data (Applicable only to some fields).
- Database normalization prevents redundancy and makes the best possible usage of storage. Proper use of foreign keys can minimise data redundancy and chance of destructive anomalies. However, concerns of efficiency and convenience can sometimes result in redundant data design despite the risk of corrupting the data.
- To overcome the Bias-Variance Tradeoff issue, introduce more data to increase the data to noise ratio which may help reduce the variance of the model. I hope this will help you in creating a better understanding of this concept. Just keep in mind, while creating any model, bias-variance tradeoff is a very important aspect to keep in mind.
- Dimensionality reduction is an important technique to overcome the curse of dimensionality. When we keep adding features without increasing the number of training samples as well, the dimensionality of the feature space grows and becomes sparser and sparser. Due to this sparsity, it becomes much easier to find a “perfect” solution for the machine learning model which highly likely leads to overfitting.
- Steps used to overcome Overfitting,
 1. Training with more data
 2. Data augmentation
 3. Data simplification
 4. Ensembling

CHAPTER 5

DATASET

5.1 Twitter Api

Social media's ubiquity has made various social media platforms more and more popular as a source of data. With this rise of social media as a data source, data collection using APIs is becoming a very sought-after skill to learn.

Twitter now has almost 400 million active monthly users , meaning a huge volume of data is available to collect, most of which is public. In addition to this, the Twitter developer team recently rebuilt the Twitter API from the ground up, releasing the Twitter API v2 in the second half of 2020. This API is really well documented and easy to use making it easier than ever to utilise this rich data source.

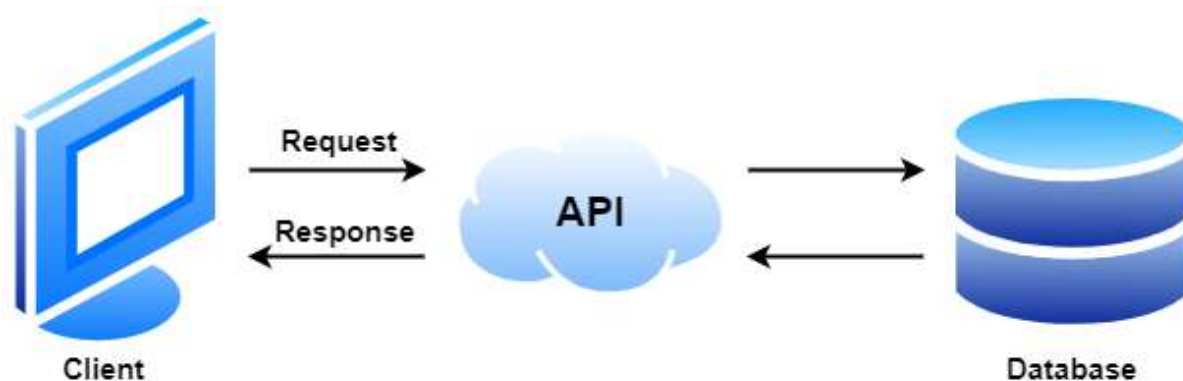


Fig 5.1 Twitter Api

5.2 Gathering Tweets

The process of applying for the standard product track and obtaining the necessary credentials involves first having a Twitter account. One must then apply for a developer account by filling out the forms on this page. Note that the recent search endpoint returns Tweets matching

the search criteria from only the last 7 days. A similar archive search endpoint `GET /2/tweets/search/all` has greater capabilities. It allows the user to return public Tweets matching a search query from as far back as Twitter's first post in 2006. However, this endpoint is only available to those users who have been approved for the Academic Research product track. The academic research product track is an exciting addition to the Twitter API v2 which allows users to use the widest range of endpoints and pull up to 10 million tweets each month! A full list of the Twitter API v2 endpoints is available [here](#). More details about features that can be used for academic research can be found [here](#). It is possible to check eligibility for this product track [here](#). The process for applying for an academic product track licence is similar, but the application form is more detailed and must be approved by the Twitter developer team.

Once the application has been approved, one can finally use various API endpoints with the standard product track. To go about this, one must set up an app by opening the developer portal and choosing 'create a new project', filling out the required details, and lastly give the new app a name. Now let's implement how to gather tweets.

5.3 Implementation

There are several ways to access data from the Twitter API in Python, for this project, we'll be using the `tweepy` python library which makes it easy to connect to and fetch data from the Twitter API. In this tutorial, we'll be fetching the tweets with a specific hashtag (`#UkraineRussiaWar`) from the API.

Source code:

```
# import tweepy
```

```
import tweepy as tw
```

Twitter API key and API secret

```
my_api_key = "f0iXIYsNGK4wEdiXX4BlOuoas"
```

```
my_api_secret = "Xj4lJl3Po0giVkcSdLLIdVhP20y2STJcwBOX3TKljmicsq7lXg"
```


authenticate

```
auth = tw.OAuthHandler(my_api_key, my_api_secret)

api = tw.API(auth, wait_on_rate_limit=True)

search_query = "#UkraineRussiaWar -filter:retweets"
```

get tweets from the API

```
tweets = tw.Cursor(api.search,

                    q=search_query,

                    lang="en",

                    since="2022-02-01").items(35000)
```

store the API responses in a list

```
tweets_copy = []

for tweet in tweets:

    tweets_copy.append(tweet)

print("Total Tweets fetched:", len(tweets_copy))

import pandas as pd
```

initialise the data frame

```
tweets_df = pd.DataFrame()
```

populate the data frame

```
for tweet in tweets_copy:

    hashtags = []

    try:

        for hashtag in tweet.entities["hashtags"]:

            hashtags.append(hashtag["text"])
```

```

text = api.get_status(id=tweet.id, tweet_mode='extended').full_text

except:

    pass

tweets_df = tweets_df.append(pd.DataFrame({'user_name': tweet.username,

                                           'user_location': tweet.user.location,\

                                           'description': tweet.user.description,

                                           'user_verified': tweet.user.verified,

                                           'date': tweet.created_at,

                                           'text': text,

                                           'hashtags': [hashtags if hashtags else None],

                                           'source': tweet.source}))

tweets_df = tweets_df.reset_index(drop=True)

```

Show the dataframe

```
tweets_df.head()
```

	user_name	user_location	description	user_verified	date	text	hashtags	source
0	BenB		Digital Sniper, Good trouble, disruption of so...	False	2022-05-07 11:00:31	The Ukrainian garrison in Mariupol can still b...	None	BenB News
1	Insights into Germany deutschland.de		or Follow us to find out more #AboutGermany - ...	True	2022-05-07 11:00:02	+++ Scholz warns of "international disorder"n...	[UkraineWar]	Sprinklr Publishing
2	Flash News	Canada	A summary of international news	False	2022-05-07 10:59:59	Protests in Russia take interesting twists htt...	[UkraineRussiaWar, UkraineWar, UkraineUnderAtt...	Twitter Web App
3	Roizgi's opinion (the voice of students)	Milan, Lombardy	Unreliable opinion on international Economy an...	False	2022-05-07 10:59:50	From @markets , the recap of the week. A lot o...	None	Twitter for iPhone
4	Nihlektra	Europe	life's a bitch and then you die or us chTo main...	False	2022-05-07 10:59:23	#Pelosi: If #Russia Is Not Listed As A State S...	[Pelosi, Russia]	Twitter for iPad

Fig 5.3 DataFrame

The above is the result of the first five columns of our dataframe.

5.4 Features

User_name - The name of the user who posted the tweet.

User_location - The location from where the user posted the tweet.

Description - The description of the user's profile.

User_verified - Boolean value (True or False) Specifying whether a user is a verified user or not.

Date - The Exact date and time as when the user posted the tweet.

Text - the actual tweet

Hashtags - hashtags used with the tweet

Source - source from which tweet was posted like web app or mobile app etc...

CHAPTER 6

ANALYSIS

6.1 Frameworks Used

- Pytorch
- Tensorflow
- Keras
- Numpy
- Sklearn
- Pandas

6.2 Insights

Let's load the data,

```
import pandas as pd
```

Storing it into a dataframe

```
tweets_df = pd.read_csv('/content/drive/MyDrive/Final/WarInUkraine_Final -WarInUkraine.csv')
```

```
tweets_df.info()
```

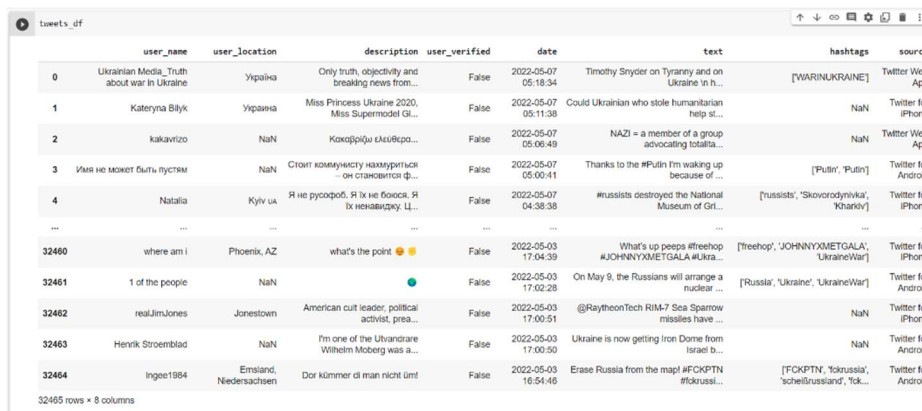
```
# Column      Non-Null Count  Dtype
---  -
0  user_name    32463 non-null  object
1  user_location 21426 non-null  object
2  description   29309 non-null  object
3  user_verified 32465 non-null  bool
4  date          32465 non-null  object
5  text          32465 non-null  object
```

6 hashtags 23347 non-null object

7 source 32465 non-null object

#the actual dataframe

Tweets_df

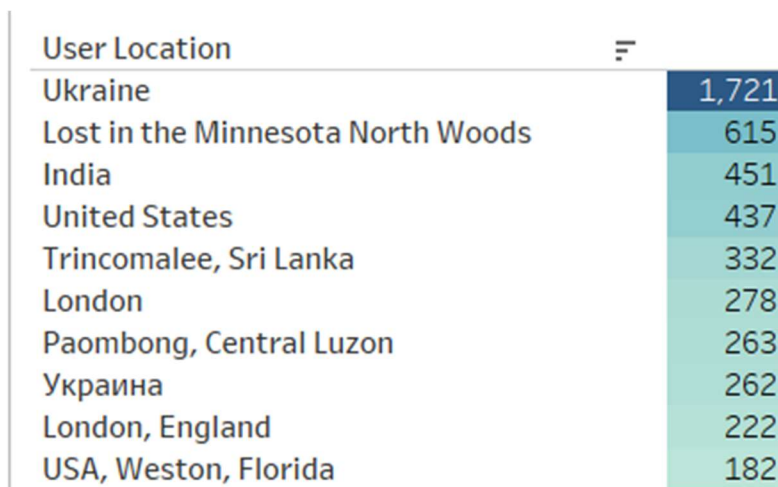


	user_name	user_location	description	user_verified	date	text	hashtags	source
0	Ukrainian Media_Truth	Ypalka	Only truth, objectivity and breaking news from...	False	2022-05-07 05:18:34	Timothy Snyder on Tyranny and on Ukraine 'n h...	[WARINUKRAINE]	Twitter Web App
1	Kateryna Bilik	Ypalka	Miss Princess Ukraine 2020, Miss Supermodel Gl...	False	2022-05-07 05:11:38	Could Ukrainian who stole humanitarian help st...	NaN	Twitter for iPhone
2	kakavizo	NaN	Кажуться Україна...	False	2022-05-07 05:06:49	NAZI = a member of a group advocating totalita...	NaN	Twitter Web App
3	Віра не має бути пустям	NaN	Стоїть комуністичу нахмуритися -- он становитися ф...	False	2022-05-07 05:00:41	Thanks to the #Putin I'm waking up because of ...	['Putin', 'Putin']	Twitter for Android
4	Natalia	Kyiv ua	Я не русофоб. Я їх не боюся. Я їх ненавижу. Ц...	False	2022-05-07 04:38:38	#russists destroyed the National Museum of Gri...	['russists', 'Skovorodnykva', 'Kharkiv']	Twitter for iPhone
...
32460	where am i	Phoenix, AZ	what's the point 🍌	False	2022-05-03 17:04:39	What's up peeps #freeshop #JOHNNYMETGALA #Ukra...	['freeshop', 'JOHNNYMETGALA', 'UkraineWar']	Twitter for iPhone
32461	1 of the people	NaN	🍌	False	2022-05-03 17:02:28	On May 9, the Russians will arrange a nuclear ...	['Russia', 'Ukraine', 'UkraineWar']	Twitter for Android
32462	realJimJones	Jonestown	American cult leader, political activist, prea...	False	2022-05-03 17:00:51	@RaytheonTech RIM-7 Sea Sparrow missiles have ...	NaN	Twitter for iPhone
32463	Henrik Stroomblad	NaN	I'm one of the Uvandrare Wilhelm Moberg was a...	False	2022-05-03 17:00:50	Ukraine is now getting Iron Dome from Israel b...	NaN	Twitter for Android
32464	Ingeer1984	Emsland, Niedersachsen	Dor kümmer di man nicht um!	False	2022-05-03 16:54:46	Erase Russia from the map! #fckPTN #fckrussi...	['fckPTN', 'fckrussia', 'schellbrussland', 'fck...']	Twitter for Android

32465 rows x 8 columns

Fig 6.2.1 Reviewed Data frame

Top 10 user locations from where most of the tweets came,



User Location	
Ukraine	1,721
Lost in the Minnesota North Woods	615
India	451
United States	437
Trincomalee, Sri Lanka	332
London	278
Paombong, Central Luzon	263
Україна	262
London, England	222
USA, Weston, Florida	182

Fig 6.2.2 Top 10 Locations

Extracting Hashtags and generating it's frequencies.

#Required libraries

```
import re
import numpy as np
import nltk
import seaborn as sns
```

```
def hashtag_extract(text_list):
    hashtags = []

    # Loop over the words in the tweet
    for text in text_list:
        ht = re.findall(r"#(\w+)", text)

        hashtags.append(ht)

    return hashtags


def generate_hashtag_freqdist(hashtags):
    a = nltk.FreqDist(hashtags)
    d = pd.DataFrame({'Hashtag': list(a.keys()),
                      'Count': list(a.values())})

    # Selecting top 15 most frequent hashtags
    d = d.nlargest(columns="Count", n = 25)

    plt.figure(figsize=(16,7))
    ax = sns.barplot(data=d, x= "Hashtag", y = "Count")

    plt.xticks(rotation=80)

    ax.set(ylabel = 'Count')
```

```
plt.show()
```

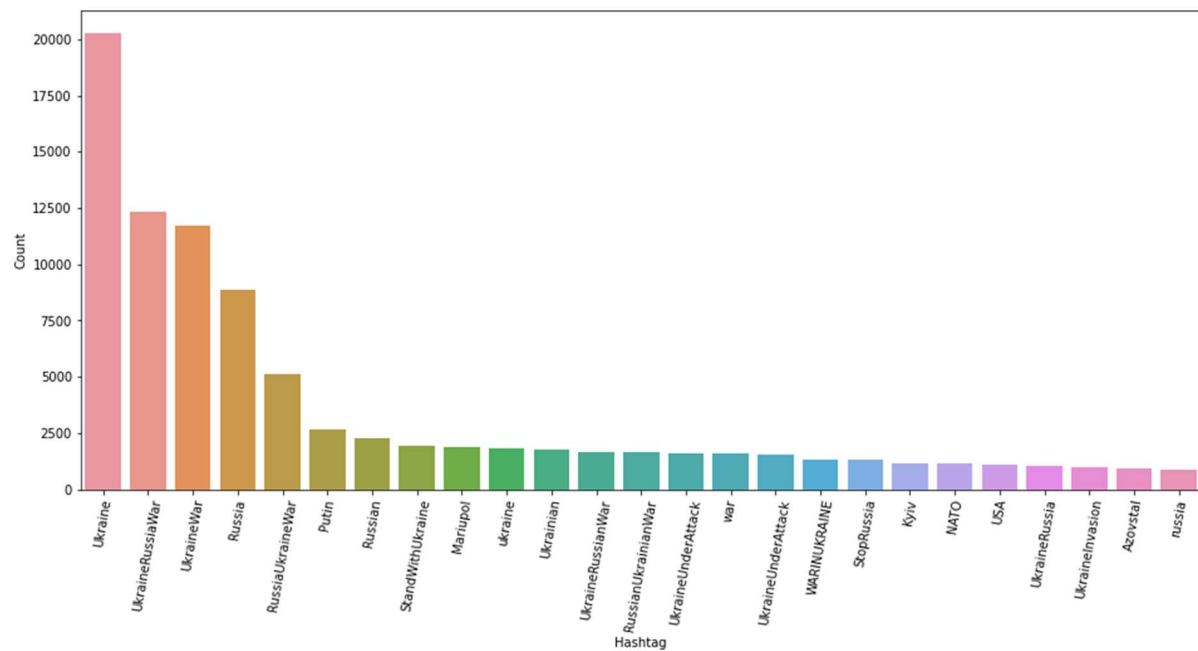
#Using our above defined function

```
hashtags = hashtag_extract(tweets_df["text"])
```

```
hashtags = sum(hashtags, [])
```

```
generate_hashtag_freqdist(hashtags)
```

And the output is as follows,



It's obvious that people are supporting Ukraine, we can see that the number of hashtags for Ukraine is higher. It shows that people are not very supportive towards Russia and it's in last place.

Data Cleaning

Preprocessing functions

```
"""
```

Removing '@names', links (http | https), Punctuations, Numbers and Special characters. Because they don't convey any sentiment of the text

```
"""
```

```
emoji_pattern = re.compile("[  
    u"\U0001F600-\U0001F64F" # emoticons  
    u"\U0001F300-\U0001F5FF" # symbols & pictographs  
    u"\U0001F680-\U0001F6FF" # transport & map symbols  
    u"\U0001F1E0-\U0001F1FF" # flags (iOS)  
    "]+", flags=re.UNICODE)
```

Basic function to clean the text

```
def clean_tweet(text):  
    text = str(text)  
  
    # Remove emojis  
    text = emoji_pattern.sub(r'', text)  
  
    # Remove identifications  
    text = re.sub(r'@\w+', '', text)  
  
    # Remove links  
    text = re.sub(r'http.?://[^\s]+[/\s]?', '', text)  
  
    return text.strip().lower()
```


The below function is used to identify the sentiment and classify them as positive, negative and neutral.

```
def analyze_sentiment(tweet):
    analysis = TextBlob(clean_tweet(tweet))
    if analysis.sentiment.polarity > 0:
        return 1
    elif analysis.sentiment.polarity <= 0:
        return 0

tweets_df['Sentiment'] = tweets_df['text'].apply(lambda x:analyze_sentiment(x))
tweets_df['Length'] = tweets_df['text'].apply(len)
tweets_df['Word_counts'] = tweets_df['text'].apply(lambda x:len(str(x).split()))
df = tweets_df[['user_location', 'description', 'Sentiment', 'user_verified',
'Length', 'Word_counts', 'text', 'hashtags']]
df.head()
df['Clean tweet'] = df['text'].apply(lambda x:clean_tweet(x))
```

After cleaning and analyzing the sentiment our data frame looks as below,

[] df.head()

	user_location	description	Sentiment	user_verified	Length	Word_counts	text	hashtags	Clean tweet
0	Україна	Only truth, objectivity and breaking news from...	0	False	82	9	Timothy Snyder on Tyranny and on Ukraine 'n h...	[WARINUKRAINE]	timothy snyder on tyranny and on ukraine 'n 7...
1	Україна	Miss Princess Ukraine 2020, Miss Supermodel Gl...	0	False	192	29	Could Ukrainian who stole humanitarian help st...	NaN	could ukrainian who stole humanitarian help st...
2	NaN	Κακαβρίζω ελεύθερα...	0	False	231	28	NAZI = a member of a group advocating totalita...	NaN	nazi = a member of a group advocating totalita...
3	NaN	Стоит коммунисту нахмуриться – он становится ф...	1	False	193	27	Thanks to the #Putin I'm waking up because of ...	['Putin', 'Putin']	thanks to the #putin i'm waking up because of ...
4	Київ ua	Я не русофоб. Я їх не боюся. Я їх ненавижду. Ці...	0	False	295	35	#russists destroyed the National Museum of Gri...	['russists', 'Skovorodnyivka', 'Kharkiv']	#russists destroyed the national museum of gri...

Distribution of sentiments,

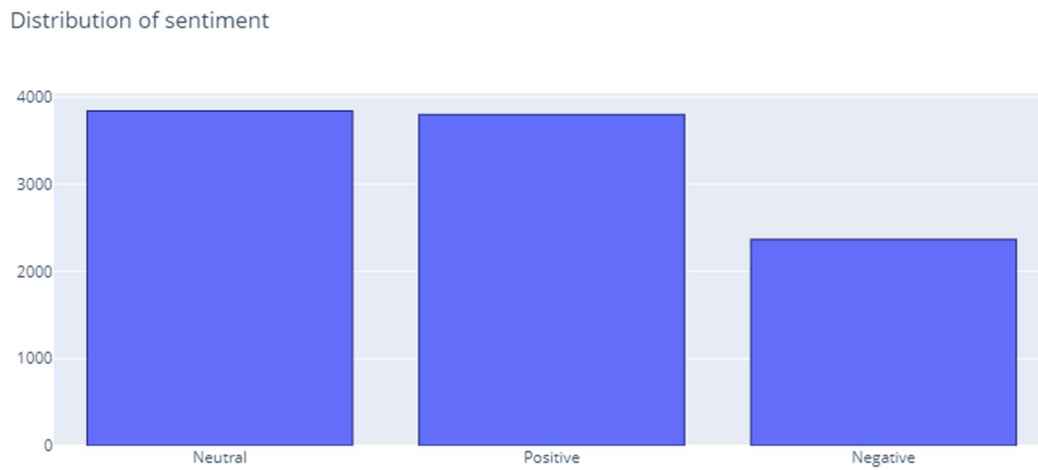


Fig 6.2.3 Distribution of sentiments

Sentiment Polarity,

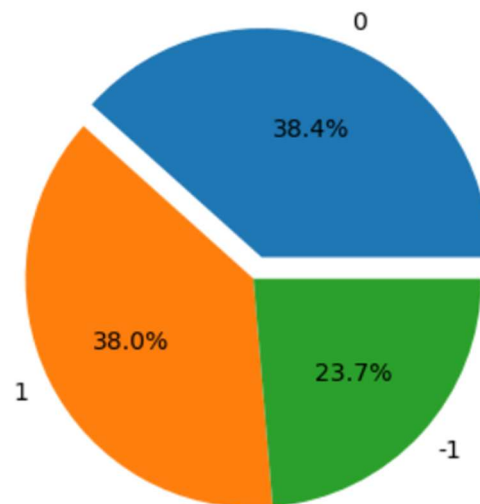


Fig 6.2.4 Sentiment Polarity

CHAPTER 7

TRAINING A NEURAL NETWORK

7.1 Train_Test_Split

Importing libraries

```
from sklearn import model_selection
```

```
X_train, X_test, y_train, y_test = model_selection.train_test_split(df['Clean tweet'],  
df['Sentiment'], test_size=0.30)
```

The clean tweet is set as observation and sentiment columns are used as target. The NumPy arrays take significantly less memory as compared to python lists. It also provides a mechanism of specifying the data types of the contents, which allows further optimisation of the code. Hence we convert our features into numpy array,

```
import numpy as np
```

```
X_train=np.array(X_train.values.tolist())
```

```
X_test=np.array(X_test.values.tolist())
```

```
y_train=np.array(y_train.values.tolist())
```

```
y_test=np.array(y_test.values.tolist())
```

Importing libraries

```
import torch
```

```
from transformers import AutoTokenizer, AutoModelForSequenceClassification
```

Downloading pre trained deep neural network model from google's server,

```
tokenizer = AutoTokenizer.from_pretrained('nlptown/bert-base-multilingual-uncased-sentiment')  
model = AutoModelForSequenceClassification.from_pretrained('nlptown/bert-base-multilingual-uncased-sentiment')
```

7.2 Bert Model

BERT Explained: State of the art language model for NLP

BERT (Bidirectional Encoder Representations from Transformers) is a recent paper published by researchers at Google AI Language. It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Question Answering (Squad v1.1), Natural Language Inference (MNLI), and others.

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.

The paper's results show that a language model which is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models. In the paper, the researchers detail a novel technique named Masked LM (MLM) which allows bidirectional training in models in which it was previously impossible.

How does BERT work?

BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT’s goal is to generate a language model, only the encoder mechanism is necessary. The detailed workings of Transformer are described in a paper by Google.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore, it is considered bidirectional, though it would be more accurate to say that it’s non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

Function to measure sentiment of each cleaned tweets,

```
def sentiment_score(review):
```

```
    tokens = tokenizer.encode(review, return_tensors='pt')
```

```
    result = model(tokens)
```

```
    return int(torch.argmax(result.logits))+1
```

```
df['sentiment_score'] = df['Clean tweet'].apply(lambda x: sentiment_score(x))
```

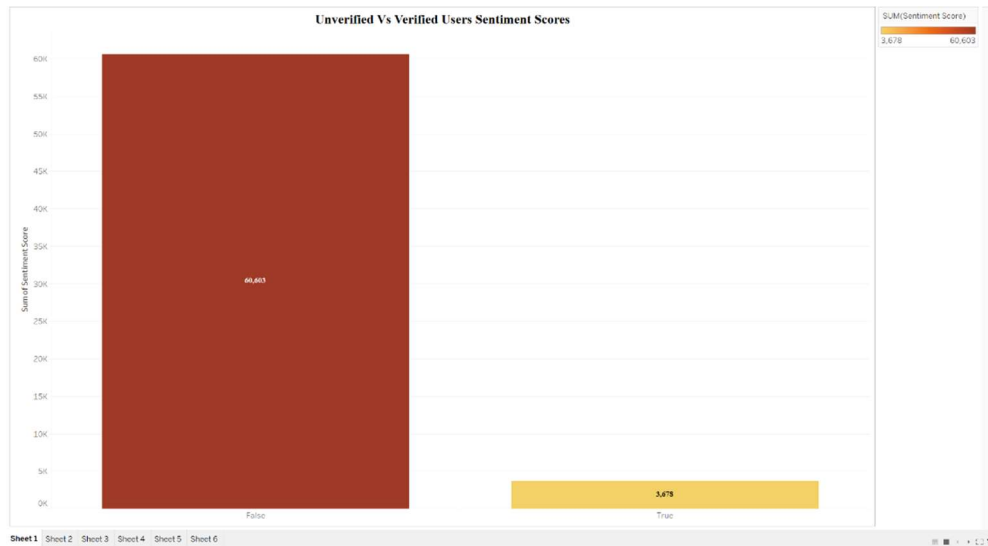
And the output looks as follows,

The Sentiment will range from 1 to 5.

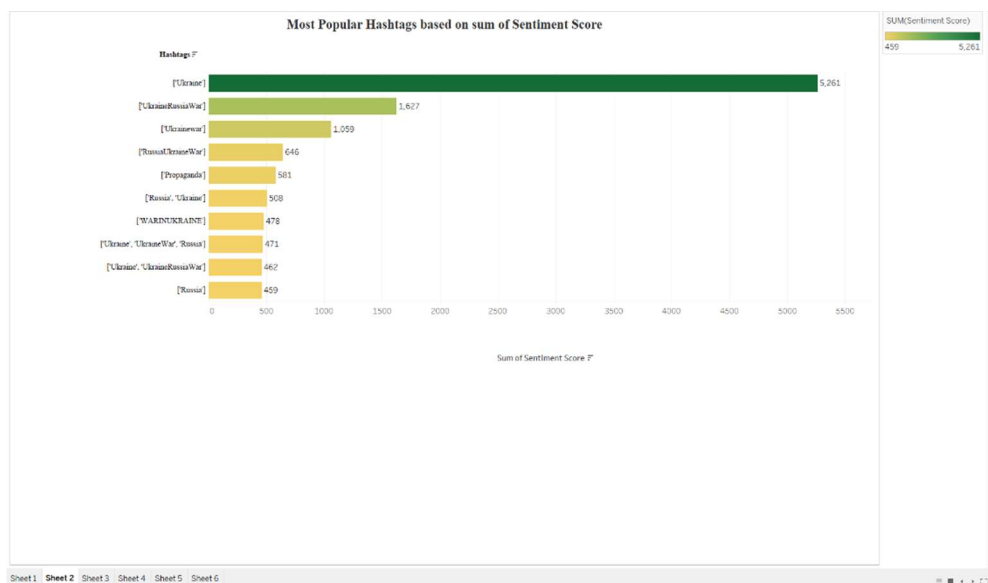
user_location	description	Sentiment	user_verified	Length	Word_counts	text	hashtags	Clean tweet	sentiment_score
Україна	Only truth, objectivity and breaking news from...	0	False	82	9	Timothy Snyder on Tyranny and on Ukraine 'n h...	['WARINUKRAINE']	timothy snyder on tyranny and on ukraine 'n 7...	1
Україна	Miss Princess Ukraine 2020, Miss Supermodel Gl...	0	False	192	29	Could Ukrainian who stole humanitarian help st...	NaN	could ukrainian who stole humanitarian help st...	1
NaN	Κακαβριζω ελευθερα...	0	False	231	28	NAZI = a member of a group advocating totalita...	NaN	nazi = a member of a group advocating totalita...	1
NaN	Стоит коммунисту нахмуриться – он становится ф...	1	False	193	27	Thanks to the #Putin I'm waking up because of ...	['Putin', 'Putin']	thanks to the #putin i'm waking up because of ...	5
Київ ua	Я не русофоб. Я їх не боюся. Я їх ненавижду. Ц...	0	False	295	35	#russists destroyed the National Museum of Gri...	['russists', 'Skovorodynivka', 'Kharkiv']	#russists destroyed the national museum of gri...	1

Insights from sentiment which we have gathered from Bert model,

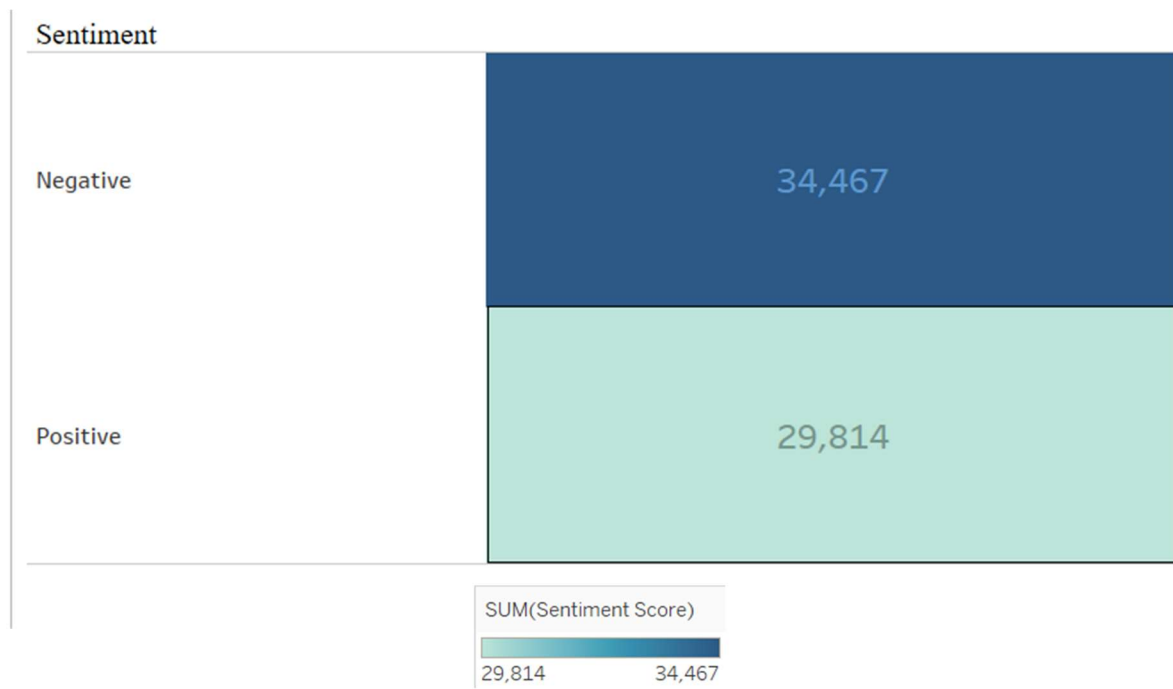
There are more unverified users hence the sum of their sentiment score is higher.



Most Popular hashtag Based on sentiment score,



Sum of sentiment scores for each sentiment,



Sequential model

A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

A Sequential model is **not appropriate** when:

- Your model has multiple inputs or multiple outputs
- Any of your layers has multiple inputs or multiple outputs
- You need to do layer sharing
- You want non-linear topology (e.g., a residual connection, a multi-branch model)

```
model.add(Embedding(voc_size, embedding_vector_features, input_length=sent_length))
model.add(Dropout(0.3))
model.add(LSTM(100)) #Adding 100 lstm neurons in the layer
model.add(Dropout(0.3))
model.add(Dense(1, activation='sigmoid'))
```

Embedding layer

Usually ML models take vectors (array of numbers) as input and, when dealing with text, we convert the strings into numbers. One of the easiest ways to do this is one-hot encoding where you treat each string as a categorical variable. But the first issue is that if you use a dictionary (vocabulary) of 10000 words, then one-hot encoding is pretty much a waste of space (memory).

Also, as discrete entities are mapped to either 0 or 1 signalling a specific category, one-hot encoding cannot capture any relation between words.

Embedding is a dense vector of floating-point values and these numbers are generated randomly and during training these values are updated via backprop just as the weights in a dense layer get updated during training.

The Embedding layer can be understood as a lookup table that maps from integer indices (which stand for specific words) to dense vectors (their embeddings).

Dropout

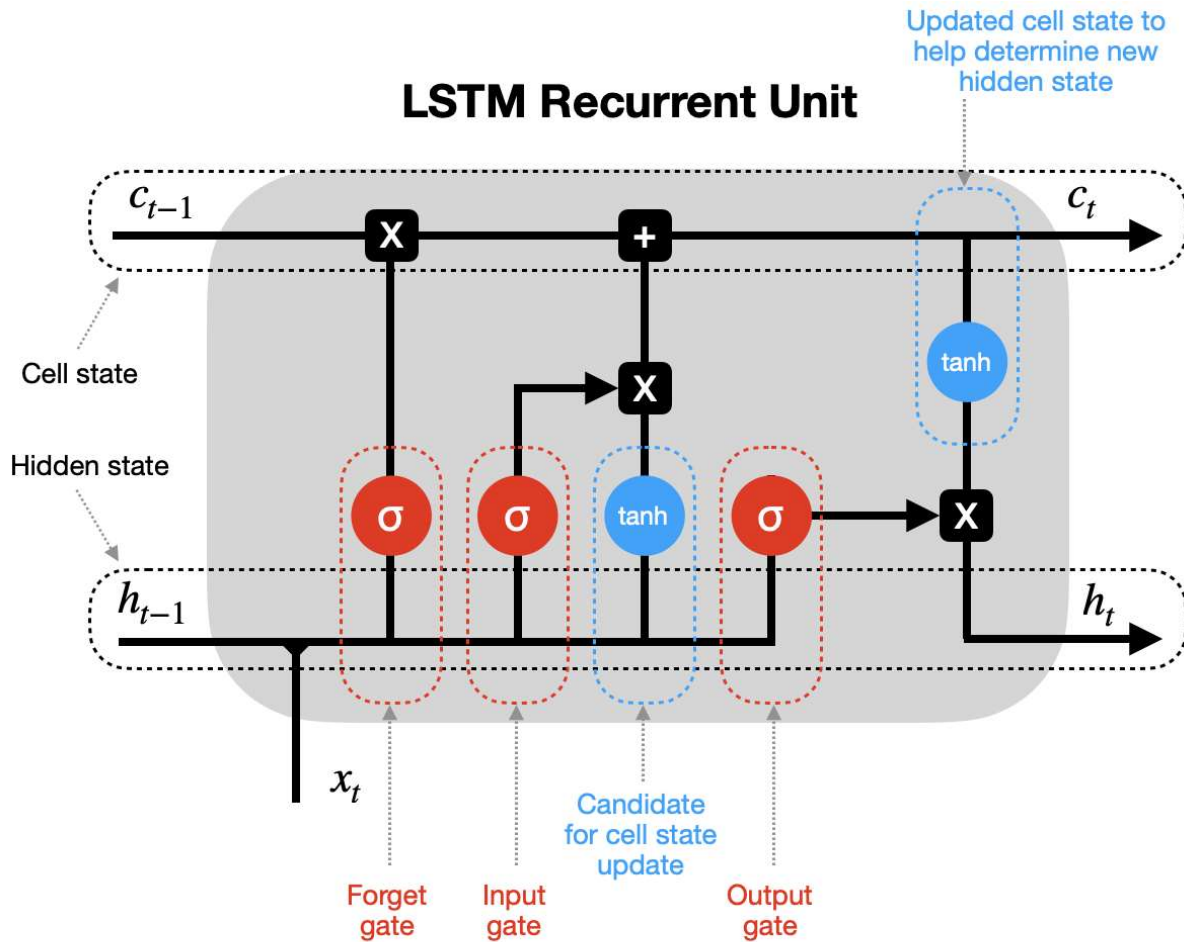
- Dropout is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly.
- This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.
- As a neural network learns, neuron weights settle into their context within the network. Weights of neurons are tuned for specific features providing some specialization.
- Neighboring neurons become to rely on this specialization, which if taken too far can result in a fragile model too specialized to the training data.
- This reliant on context for a neuron during training is referred to complex co-adaptations.
- You can imagine that if neurons are randomly dropped out of the network during training, that other neurons will have to step in and handle the representation required to make predictions for the missing neurons.
- This is believed to result in multiple independent internal representations being learned by the network.
- The effect is that the network becomes less sensitive to the specific weights of neurons. This in turn results in a network that is capable of better generalization and is less likely to overfit the training data

7.3 LSTM

The LSTM recurrent unit is much more complex than that of RNN, which improves learning but requires more computational resources.

Let's go through the simplified diagram (weights and biases not shown) to learn how LSTM recurrent unit processes information.

1. Hidden state & new inputs — hidden state from a previous timestep (h_{t-1}) and the input at a current timestep (x_t) are combined before passing copies of it through various gates.
2. Forget gate — this gate controls what information should be forgotten. Since the sigmoid function ranges between 0 and 1, it sets which values in the cell state should be discarded (multiplied by 0), remembered (multiplied by 1), or partially remembered (multiplied by some value between 0 and 1).
3. Input gate helps to identify important elements that need to be added to the cell state. Note that the results of the input gate get multiplied by the cell state candidate, with only the information deemed important by the input gate being added to the cell state.
4. Update cell state —first, the previous cell state (c_{t-1}) gets multiplied by the results of the forget gate. Then we add new information from [input gate \times cell state candidate] to get the latest cell state (c_t).
5. Update hidden state — the last part is to update the hidden state. The latest cell state (c_t) is passed through the tanh activation function and multiplied by the results of the output gate.



h_{t-1} - hidden state at previous timestep t-1 (short-term memory)

c_{t-1} - cell state at previous timestep t-1 (long-term memory)

x_t - input vector at current timestep t

h_t - hidden state at current timestep t

c_t - cell state at current timestep t

X - vector pointwise multiplication **+** - vector pointwise addition

tanh - tanh activation function

σ - sigmoid activation function

T - concatenation of vectors

- states

- gates

- updates

importing necessary libraries

```
torch.backends.cudnn.deterministic = True

import matplotlib.pyplot as plt

from keras.models import Sequential, load_model

from keras.layers import Dense, LSTM, Embedding, Dropout

from keras.preprocessing.text import Tokenizer

from keras.preprocessing.sequence import pad_sequences
```

Checking whether the notebook is connected to gpu

```
DEVICE = torch.device('cuda:1' if torch.cuda.is_available() else 'cpu')

print (DEVICE)

#prints → cuda:1
```

creating a tokenizer and padding our text vector so that they all have the same length

```
tokenizer = Tokenizer (num_words=5000, split=" ")

tokenizer.fit_on_texts(data['Clean tweet'].values)

X = tokenizer.texts_to_sequences(data['Clean tweet'].values)

X = pad_sequences(X)

X[:5]
```

creating a sequential LSTM model with dense layers

```
model = Sequential()

model.add(Embedding(5000, 256, input_length=X.shape[1]))

model.add(Dropout(0.3))

model.add(LSTM(256, return_sequences=True, dropout=0.3, recurrent_dropout=0.2))

model.add(LSTM(256, dropout=0.3, recurrent_dropout=0.2))

model.add(Dense(3, activation='softmax'))
```

compiling our model and its summary is as follows,

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 76, 256)	1280000
dropout (Dropout)	(None, 76, 256)	0
lstm (LSTM)	(None, 76, 256)	525312
lstm_1 (LSTM)	(None, 256)	525312
dense (Dense)	(None, 3)	771

Total params: 2,331,395

Trainable params: 2,331,395

Non-trainable params: 0

Splitting and training our model

```
y = pd.get_dummies(data['Sentiment_text']).values
```

```
[print(data['Sentiment_text'][i], y[i]) for i in range(0,5)]
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Fitting our model with eight epochs

```
batch_size = 32
```

```
epochs = 8
```

```
model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size, verbose=2)
```

Epoch 1/8

812/812 - 649s - loss: 0.5044 - accuracy: 0.8057 - 649s/epoch - 800ms/step

Epoch 2/8

812/812 - 637s - loss: 0.2382 - accuracy: 0.9303 - 637s/epoch - 785ms/step

Epoch 3/8

812/812 - 634s - loss: 0.1962 - accuracy: 0.9421 - 634s/epoch - 780ms/step

Epoch 4/8

812/812 - 636s - loss: 0.1629 - accuracy: 0.9511 - 636s/epoch - 783ms/step

Epoch 5/8

812/812 - 639s - loss: 0.1321 - accuracy: 0.9591 - 639s/epoch - 786ms/step

Epoch 6/8

812/812 - 637s - loss: 0.1086 - accuracy: 0.9661 - 637s/epoch - 785ms/step

Epoch 7/8

812/812 - 634s - loss: 0.0851 - accuracy: 0.9731 - 634s/epoch - 781ms/step

Epoch 8/8

812/812 - 631s - loss: 0.0665 - accuracy: 0.9794 - 631s/epoch - 777ms/step

<keras.callbacks.History at 0x7fa660a36a10>

The model achieved 97 percent accuracy.

Predicting with test data

```
predictions = model.predict(X_test)
```

```
[print(data['Clean tweet'][i], predictions[i], y_test[i]) for i in range(0, 5)]
```

7.4 Prediction

#Predictions →

- could ukrainian who stole humanitarian help still called ukrainian and could not russian be blamed for being russian if he sent billions for help to ukraine war in ukraine ukraine ukraine war [3.0008616e-04 3.4888420e-05 9.9966502e-01] [0 0 1].
- nazi a member of a group advocating totalitarian government seeking territorial expansion antisemitist believe in his race supremacy leading to war and genocide putin putinwarcriminal warinukraine sh8a516sty [5.6100974e-04 4.1224212e-05 9.9939775e-01] [0 0 1].
- thanks to the putin im waking up because of the shillings and siren he liberated me from a normal life putin war criminal war ukraine stand with ukraine ukraine war russian russia [0.0023376 0.99510443 0.00255793] [0 1 0].
- russists destroyed the national museum of grigoriy skovoroda in skovorodynivka village kharkiv region on the night of may bastards want to destroy everything connected with our culture and memory save ukrainian history ty q8 bercs [1.8479455e-04 9.9938977e-01 4.2538816e-04] [0 1 0] [None, None, None, None, None]

#Prediction result on test dataset

pos_count, neu_count, neg_count = 0, 0, 0

real_pos, real_neu, real_neg = 0, 0, 0

for i, prediction in enumerate(predictions):


```

if np.argmax(prediction)==2:

    pos_count += 1
elif np.argmax(prediction)==1:

    neu_count += 1
else:

    neg_count += 1
if np.argmax(y_test[i])==2:

    real_pos += 1
elif np.argmax(y_test[i])==1:

    real_neu += 1
else:

    real_neg +=1


print('Positive predictions:', pos_count)
print('Neutral predictions:', neu_count)
print('Negative predictions:', neg_count)
print('Real positive:', real_pos)
print('Real neutral:', real_neu)
print('Real negative:', real_neg)

```

#Prints →

Positive predictions: 2432

Neutral predictions: 2674

Negative predictions: 1387

Real positive: 2387

Real neutral: 2627

Real negative: 1479

Finally Our implemented model architecture can be visualised with keras visualiser,

```
!pip3 install keras_visualizer
```

```
from keras_visualizer import visualizer
```

```
from keras.utils.vis_utils import plot_model
```

```
model = load_model('sentiment_analysis.h5')
```

```
plot_model(model)
```

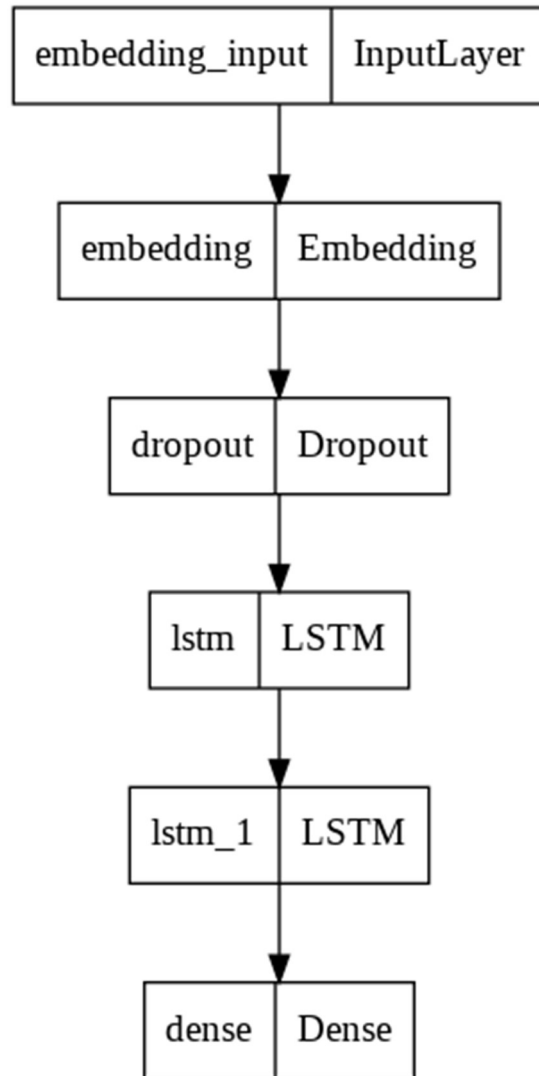


Fig 7.4 Model Layers

CHAPTER 8

CONCLUSION

From the above analysis we can conclude that People are supporting ukraine,

When sentiment is positive the sentiment score in top 10 countries are,

User Location	Count of User Location	Sentiment Score
Ukraine	685	1,569
India	174	375
United States	168	438
USA, Weston, Florida	166	422
Lost in the Minnesot..	146	253
Paombong, Central L..	98	123
London	92	238
Украина	84	206
London, England	72	164
Trincomalee, Sri Lan..	68	96

When sentiment is negative the sentiment score in top 10 countries are,

User Location	Count of User Location	Sentiment Score
Ukraine	1,036	1,742
Lost in the Minnesot..	469	564
India	277	493
United States	269	501
Trincomalee, Sri Lan..	264	308
London	186	308
Украина	178	259
Paombong, Central L..	165	173
London, England	150	248
USA, Weston, Florida	16	16

REFERENCES

[1] Stock Prediction with Stacked-LSTM Neural Networks Authors : Xiaochun Zhang; Chen Li; Kuan-Lin Chen; Dimitrios Chrysostomou; Hongji Yang

[2] Sentiment Analysis Pushpak Bhattacharyya
Indian Institute of Technology Delhi, New Delhi, Delhi, IN

[3] Sentiment Analysis Is a Big Suitcase by Erik Cambria