9

by 9 9

Submission date: 31-May-2022 05:53PM (UTC+0530)
Submission ID: 1847781149
File name: DMA_Co19322_Final_Report.docx (1.89M)
Word count: 4611
Character count: 27260

Chandigarh College of Engineering and Technology
(Degree Wing)

Compiler Design (CS – 605C)
Final Project File

Submitted By
Dipesh Singla
Roll: - CO19322
C.S.E 6th Semester

Submitted To
Dr. Varun Gupta
Professor CSE, HOD (A.S)

Table of Contents

S.No	Topics	Page No.
1.	Problem Introduction	3
2.	Project methodology	3
3.	Flow of Work	4
	3.1 ETL(Exploration, Transformation and Loading)	4
	3.2 Feature Engineering	4
	3.3 Model Definition and Training (Machine Learning Models)	4
	3.4 Model Definition and Training (Deep Learning Algorithms)	4
	3.5 Model Evaluations	
4.	Architectural Decisions/ Tools Used	4
	4.1 Data Source	5
	4.2 Enterprise Data	5
	4.3 Data Repository	5
	4.4 Discovery and Exploration	5
	4.5 Actionable Insights	5
	4.6 Applications / Data Products	5
	4.7 Security, Information Governance and Systems Management	6
5.	Final Results	
	5.1 Based on the Accuracy of the chosen model	
	5.2 Based on the F1 scores of the chosen model	
	5.3 Table of Comparison	
6.	Conclusion	9
7.	References	9
8.	Source Code	10-30

1. Problem Introduction

News is being circulated in an ever-increasing social context, whether on social media platforms or the internet and that too in bulk. With so much news arriving from everywhere, it becomes difficult to check the source and validity of the news, that is, to distinguish between false and accurate news. This project provides an overview of news detection, which uses existing Machine Learning Models and Deep Learning Algorithms to determine whether the news is correct, i.e. authentic, and trustworthy.

Why fake news is a problem?

Fake news is defined as misinformation, disinformation, or malicious information conveyed by word of mouth and conventional media, as well as, more recently, digital modes of communication such as manipulated videos, memes, unconfirmed adverts, and social media promoted rumours. Fake news on social media has become a severe concern, with the potential for it to lead to mob violence, suicides, and other negative outcomes as a result of disinformation spreading on social media.

Keywords
True and Fake News, Data Mining, Twitter, Google News Vector, word2vecgensim model

2. Project Methodology

Clement Bisaillon at [1] has donated the dataset used. It comprises two CSV files, one for bogus news called Fake.csv and the other for factual news called True.csv.

First, we displayed each of the data columns (such as Title, Subject, and Date) and examined how they are connected and what sort of problem we can deduct from that situation. And concluded that it is a binary classification problem in which a news item can be either phony or truthful. Using these representations, we vectorized the textual data in several ways and predicted it using the Logistic Regression Model, Decision Tree Classifier, and Artificial Neural Network. The Decision Tree Classifier surpasses other models in identifying false and real with an accuracy of 99.63 percent in the final model tests. As of now, a comprehensive solution for the use case is also attached.

3. Flow of Work

The flow of work is as shown:

3.1 ETL (Exploration, Transformation, and Loading)

a) Analysis of Dates
b) Analysis of Subjects
c) Analysis of Title
- Based on the length of the title
- Based on the word count of the title
- Visualizing the most used words (100) (for this we used df_data_1 and df_data_2)

d) Analysis of Text
- Based on the length of text
- Based on word count of column text
- Visualizing the most used words (100) (for this we used df_data_1 and df_data_2)

3.2 Feature Engineering

a) Data Cleaning
b) Pre-Processing
c) Feature Creation
d) Visualizations

3.3 Model Definition and Training (Machine Learning Models)

a) Using Logistic Regression (from scikit-learn) with 96D vector features provided by spaCy
b) Using Decision Tree Classifier (from sci-kit-learn) with 96D vector features provided byspaCy
c) Iteration/Approach 1: Using the above models mentioned with spaCy vectors as inputfeatures for predictions.
d) Iteration/Approach 2: Using the above models mentioned by adding CountVectorizer andTFIDF Transformer from sklearn.feature_extraction

3.4 Model Definition and Training (Deep Learning Algorithms)

a) Using Artificial Neural Network from a Sequential Model
b) Iteration 2 uses more features with accuracy and F1-score as the primary metrics of evaluation.

3.5 Model Evaluations

a) Based on test accuracy
b) Based on F1 scores

4. Architectural Decisions/ Tools Used

4.1 Brief description of the dataset

This dataset consists of about 40000 articles consisting of fake as well as real news. We aim to train our model so that it can correctly predict whether a given piece of news is real or fake. The fake and real news data are given in two separate datasets with each dataset consisting of around 20000 articles.

4.2 Data Source

a) **Technology Choice**:

The data source makes data available in CSV format as Fake.csv and True.csv [1].

b) **Justification:**

It is simple to access, gather and explore such datasets, and they also includeresearch on a variety of people, which provides us with the foundational approach for working on machine learning or deep learning challenges.

4.3 Enterprise Data

a) **Technology Choice:** Not required initially. We can use Lift for the enterprise data if required.

b) **Justification:** The lift enables us to efficiently shift on-premises data to cloud databases. In our situation, we used a preset data set and verified that it worked in any similar real-world scenario.

4.4 Data Repository

a) **Technology Choice**: Cloud Storage (IBM)

b) **Justification**: Cloud object storage allows you to store almost unlimited amounts of data. It's commonly utilized for scalable and long-term data archiving and backup, online and mobile apps, and analytics storage.

4.5 Discovery and Exploration

a) **Technology Choice:** Python, Jupyter, sci-kit-learn, pandas, Matplotlib, Seaborn

b) **Justification**: Open source and supported in IBM Cloud are Jupyter, Python, sci-kit-

learn, pandas, Matplotlib, and Seaborn. Some of these components have overlapping characteristics, while others have complementing characteristics.

4.6 Actionable Insights

a) **Technology Choice:** Python, pandas, nltk, and sci-kit-learn for Machine Learning Models and Keras and Tensorflow for Deep Learning Algorithm

b) **Justification**: Python, pandas, nltk, and sci-kit-learn are great alternatives to R/R-Studio for data research. Python is also a cleaner and simpler programming language to learn than R. Pandas is Python's equivalent of R data frames, giving access to relational data. Nltk is a natural language processing toolkit that comes in handy when dealing with vast quantities of text. Finally, scikit-learn conveniently organizes all of the machine learning methods necessary. It is also available on IBM Cloud via IBM Watson Studio.

Keras and TensorFlow are the most popular technologies for deep learning algorithms for framing neural networks. One of the most commonly used deep learning frameworks is TensorFlow. It is a linear algebra library that supports automated differentiation at its core. TensorFlow's Python-based syntax is rather complicated. As a result, Keras adds an abstraction layer to TensorFlow. Watson Studio and Watson Machine Learning on IBM Cloud provide seamless support for both frameworks.

4.7 Applications / Data Products

a) **Technology Choice**: Export of an already run, static Jupyter Notebook

b) **Justification**: IBM Watson Studio's Jupyter Notebook is one of the simplest, fastest, and most efficient methods to provide a data product based on what we have worked on. It not only provides a fantastic platform for doing everything at once, but it also keeps everyone on the same page, whether it's data visualization, cleaning, pre-processing, training the model, or delivering the entire project.

4.8 Security, Information Governance and Systems Management

a) **Technology Choice**: Cloud Object Storage

b) **Justification:** When it comes to contemporary, cost-effective cloud storage, Object Storage is the gold standard.

5. **Proposed approach/ Methodology and Model**:

In this work, in Phase-1 first of all we've cleaned the data, then a list of indices where the publisher is not mentioned, then Separating Publication info, the from the actual, text and The listing text column with new text there were 630 rows in fake news with empty text, then we created world cloud for both true and fake news and both were pretty different. In pre-processing text we first combined the title and text for real and fake news. Then in Phase-2 we used vectorization in Word2Vec, Word2Vec is one of the most popular technique to learn word embeddings using shallow neural network. Then we created Word2Vec model with genism. Then we feeded US presidents, etc to form some vectors. Further we tokenized the text and Checking the first 10 words of first news # every word has been represented with a number, made Histogram for no of words in news shows that most news article are under 700 words, and created a weight matrix. Then embedding vectors from word2vec and usings it as weights of non-trainable keras embedding layer and defined Neural Network. Then finally predict probability of news being real so converting into classes.

Lastly we used pre-trained Word2Vec explored them and got the required accuracies and F-1 Scores.

6. **Block Diagrams**:

Block Diagram for fake news detection in Previous research works.

1. **Framework Block-Diagram of Fake News detection with NLP and ML**.

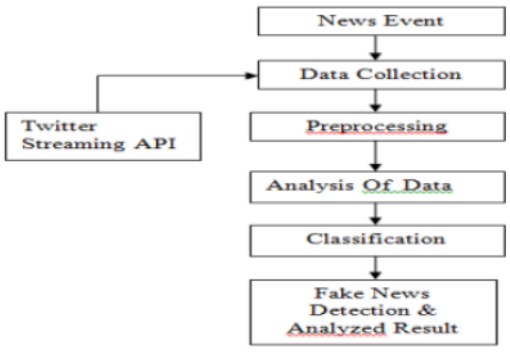


Figure 1: Framework Block-Diagram of Fake News detection with NLP and ML [6].

2. Fake News detection using supervised Learning Algorithms

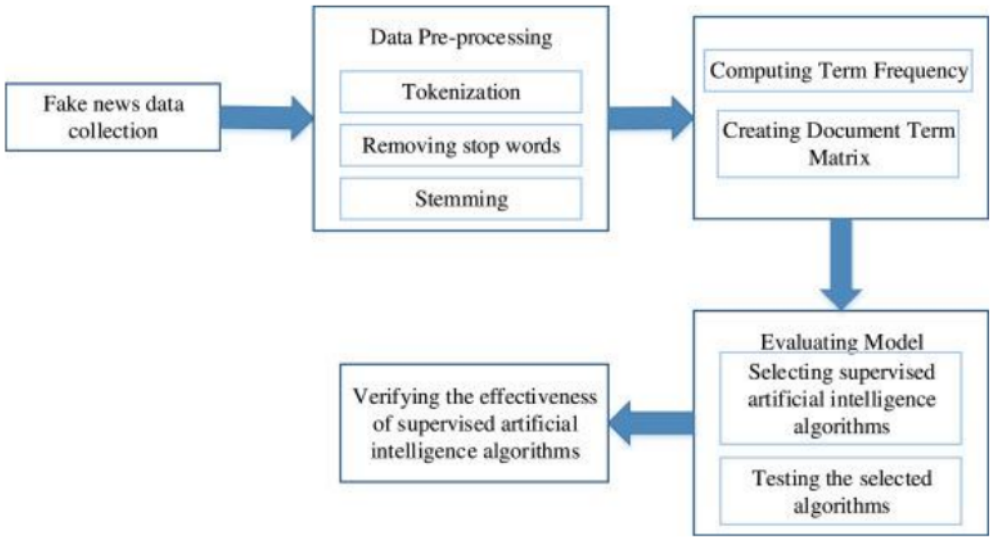


Figure 2: Fake News detection using supervised Learning Algorithms [7].

3. MBi-LSTM method in the Fake news detection method

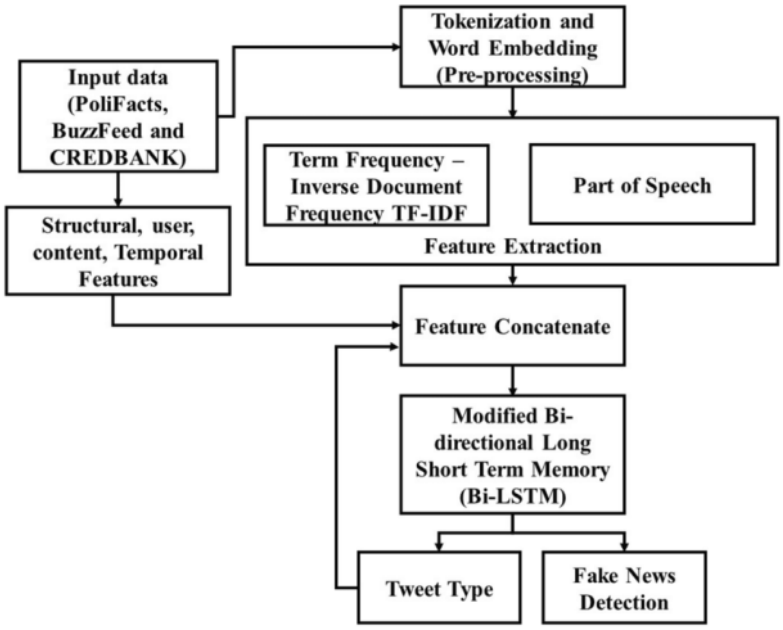


Figure 3: Block diagram of MBi-LSTM method in the Fake news detection method [8].

4. Fake news detection approaches:

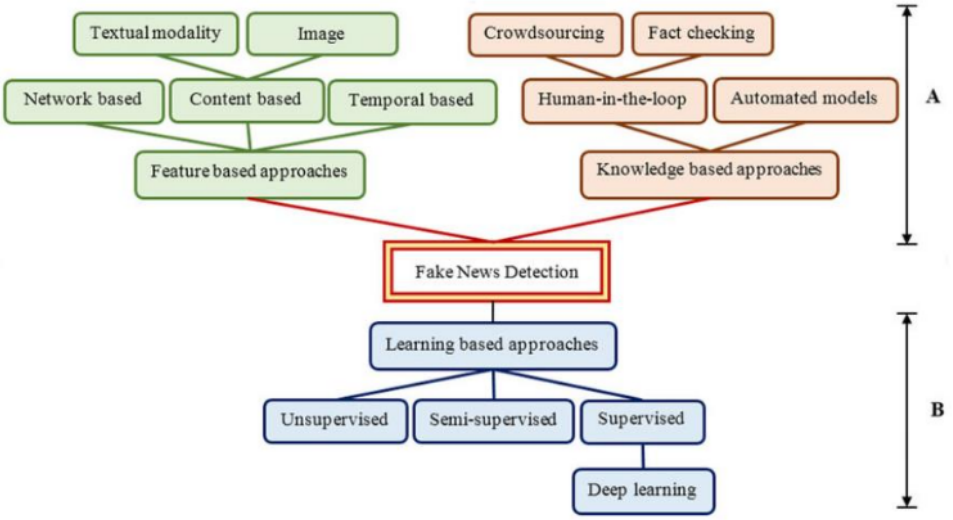


Figure 4: Fake news detection approaches [9]

5. Development of a smart system for fake news detection

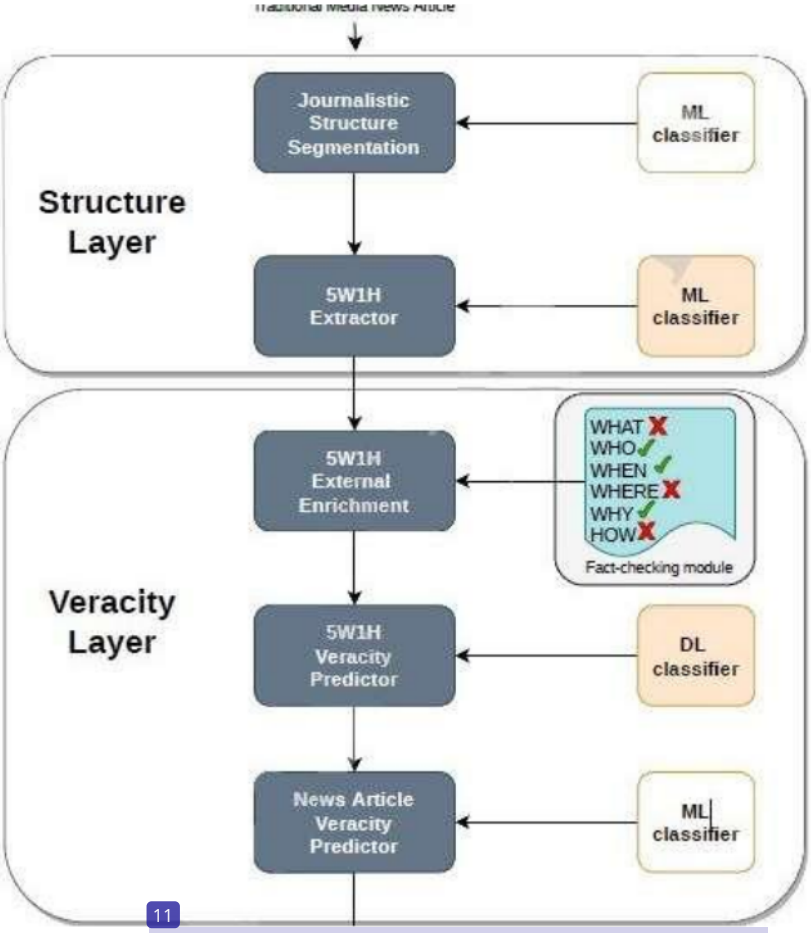


Figure 5: Development of a smart system for fake news detection [10]

6. **EchoFakeD: improving fake news detection in social media with an efficient deep neural network**

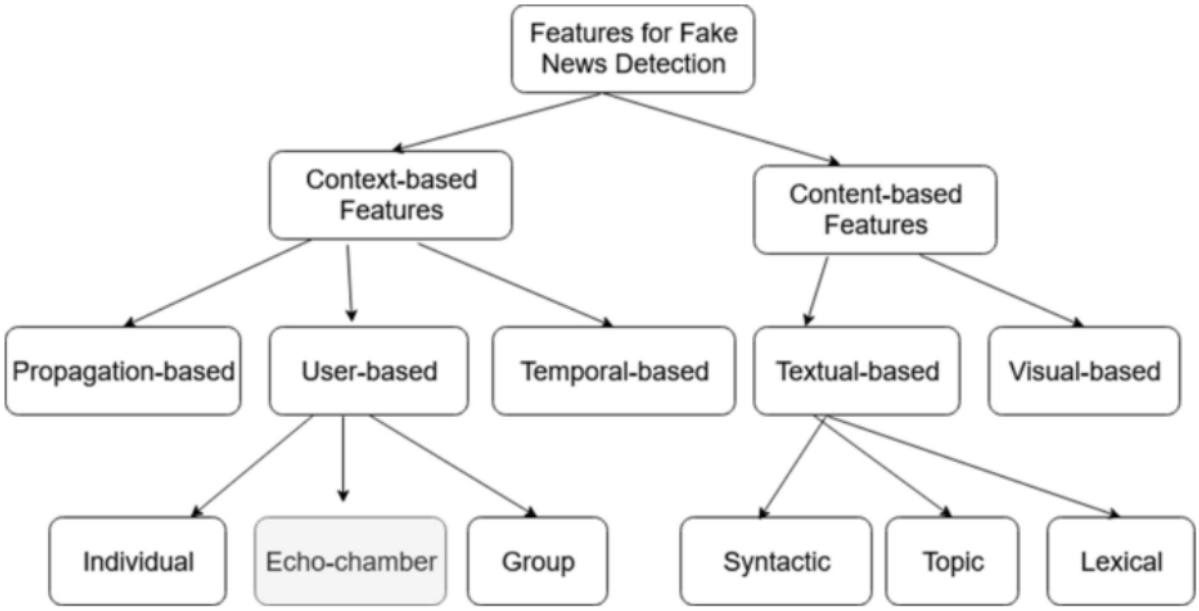


Figure 6: Features of Fake news detection [11]

7. Final Results

Along with a comparison from other top models Based on the chosen model's accuracy.

1. By selecting blogs as the solver and increasing the maximum number of iterations to 200, we can get an accuracy of 90.77 percent on the testing data for Logistic Regression with spaCy vectors as input features.
2. With spaCy vectors as input features, the decision tree classifier achieves an accuracy of 85.51 percent.
3. Logistic Regression with textual data as an input feature and Count Vectorizer andTFIDF Transformer in pipeline achieves 98.93% accuracy.
4. A Decision Tree Classifier using textual data as an input feature and Count Vectorizer and TFIDF Transformer in the pipeline achieves 99.63 percent accuracy.
5. Using 1000 input features, an artificial neural network constructed with Keras and TensorFlow backend with the Sequential Model and dense layers achieves an accuracy of 99.28 percent.
6. An Artificial Neural Network built using Keras and Tensorflow backends with the Sequential Model and dense layers achieves an accuracy of 99.21 percent on testing data with 10000 input features. Based on the F1 scores of the selected model, we can calculate the number of false positives and false negatives. Overall, we get an accuracy of 98.93%.
7. By selecting blogs as the solver and increasing the maximum number of iterations to 200, we can obtain a f1-score of 90.42 percent on the testing data for Logistic Regression with spaCy vectors as input features.
8. A decision tree classifier with spaCy vectors as input features achieves an f1-score of 84.49 percent.

9. Logistic Regression with textual data as an input feature and Count Vectorizer andTFIDF Transformer in the pipeline yields an f1 of 98.81%.
10. A Decision Tree Classifier using textual data as an input feature and Count Vectorizer and TFIDF Transformer in the pipeline produces an f1-score of 99.61 percent.
11. Keras with Tensorflow backend Artificial Neural Network with Sequential Model and Dense Layers supply us with an f1-score of 99.25 percent on the testing data with1000 input features.
12. An artificial neural network built using Keras and Tensorflow backends with the Sequential Model and dense layers achieve a f1-score of 99.17 percent on testing data with 10000 input features. Also, an F1 score comparable to that of accuracy demonstrates how accurate our model is in forecasting

Future Scope:

1. Gathering specific news-related tweets.
2. Using the same Decision Tree Classifier and pipeline, we will forecast whether ornot this is false news.
3. Another suggestion is to utilize the same for the credibility score assignment of acertain tweet, which is slightly different from this.

Table of Comparison

Table of comparison of fake news detection and clean is shown in Table 1.

Model/ Classifier	Epochs/ Iterations	Batch Size	Input Dimensions	Test Set Accuracy	Test Set F1-Score	Other Parameters
Logistic Regression	200		spaCy Vectorized Features	90.77	90.42	solver = lfbgs
Logistic Regression	200		Features from Count Vectorizer	98.93	98.88	solver = lfbgs
Decision Tree Classifier	-		spaCy Vectorized Features	85.51	84.49	criterion=entropy max_depth = 10 splitter=best, random_state=2020

Decision Tree Classifier	-		Features from Count Vectorizer	**99.63**	99.20	criterion=entropy max_depth =10 splitter=best random_state=2020
Neural Network 1	7	512	1000 Features from Count Vectorizer	99.19	99.16	optimizer=adam loss=binary_crossentropy
Neural Network 2	7	512	10000 Features from Count Vectorizer	99.24	99.12	optimizer=adam loss=binary_crossentropy

8. Conclusion

Using Pre-Trained Word2Vec Vectors in a pipeline with we've achieved maximum accuracy, recall, and F1 score of about 99%, but there is always room for improvement. Various things, such as tweaking the hyperparameters, can be done in the future to improve the accuracy and f1 score based on another dataset. The use of tensors mayenhance the overall results. Since the beginning of the project, the authors were enthralled while working, facing different challenges: may it be in terms of preprocessing the textual data, model training, or evaluation. Finally, the maximum accuracy and F1 score as per the models trained are achieved by using the Decision Tree Classifier in a pipeline with the Count Vectorizer and TFIDF Transformer, while with other models the team believes that there is always a scope of improvement accommodating various changes. Pretrained 300-D spacy model (en_core_web_lg()) can be used which can take the models having a lower accuracy to the next level, on the contrary, it will require a huge amount of system resources to be compromised. Nonetheless, more hyperparameter tuning can be done in other prospects depending on another dataset using, this project will help us in our future research inin which collection of certain news-based tweets can be performed using twtwinsnd then using the same Decision Tree Classifier with the said pipeline, it can predict whetherthat is a piece of fake news or not. Apart from this, another idea is to use the same for the credibility score assignment of a particular tweet. As a whole, the project will help the contributors build models or research.

9. References

1. Dataset Link: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset
2. Ahmed H, Traore I, Saad S. “Detecting opinion spams and fake news using text classification”, Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.
3. Ahmed H, Traore I, Saad S. “Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham (pp. 127-138).
4. Fake News Vs Real News Dataset
5. The lightweight IBM Cloud Garage Method for data science
6. Dataset Link: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset
7. Ahmed H, Traore I, Saad S. “Detecting opinion spams and fake news using text classification”, Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.

8. Ahmed H, Traore I, Saad S. "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham (pp. 127-138).
9. Fake News Vs Real News Dataset
10. The lightweight IBM Cloud Garage Method for data science
11. Ahmed H, Traore I, Saad S. "Detecting opinion spams and fake news using text classification", Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.
12. Ahmed H, Traore I, Saad S. (2017) "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham (pp. 127-138).
13. Ahmed H, Traore I, Saad S. "Detecting opinion spams and fake news using text classification", Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.
14. Ahmed H, Traore I, Saad S. (2017) "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham (pp. 127-138).

Appendix:

Source Code

Data Mining Lab:2019-2023 Batch

Semester - 6: Final Project: True vs Fake and Cleaning News DetectionName: Dipesh

Singla Roll: CO19322

Keywords: True and Fake News, Data Mining, Twitter, Google News Vector, word2vecgensim model

WHY FAKE NEWS IS A PROBLEM?

Fake news is defined as misinformation, disinformation, or mal-information that spreads by word of mouth, conventional media, and, more recently, digital modes of communication such as manipulated films, memes, unconfirmed adverts, and social media disseminated rumors. Fake news on social media has become a severe concern, with the potential for it to lead to mob violence, suicides, and other negativeoutcomes as, a result of disinformation propagated on social media.

Motivation

In this ever-increasing social world, we see lots of news circulating may it be on social media platforms or the internet across the globe. So there is a need to classify how can one be sure whether the news is true i.e. is genuine and can be trusted up by existing Machine Learning Models and Deep Learning Algorithms. We have also worked on research on how various social

networking sites like Twitter make their algorithms work to get to know the facts about particular news which helped us carry this idea forward.

Use Case: To visualize, classify, predict and compare various models/ preprocessing algorithms to check whether the news is true or fake based on a given dataset.

Data Sets

1. https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset
2. https://www.kaggle.com/datasets/sandreds/googlenewsvectorsnegative300

Library Imports

```
import warnings warnings.filterwarnings('ignore')

import numpy as np import
pandas as pd
import matplotlib.pyplot as pltimport seaborn as
sns
import nltkimport re


from wordcloud import WordCloudfrom tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequencesfrom tensorflow.keras.models
import Sequential


from tensorflow.keras.layers import Dense, Embedding, LSTM, Conv1D,
MaxPool1D
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
```

Exploring Fake News

```
fake =
pd.read_csv("/kaggle/input/fake-and-real-news-dataset/Fake.csv")

fake.head()
```

	title	text	subject	date
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017
3	Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017

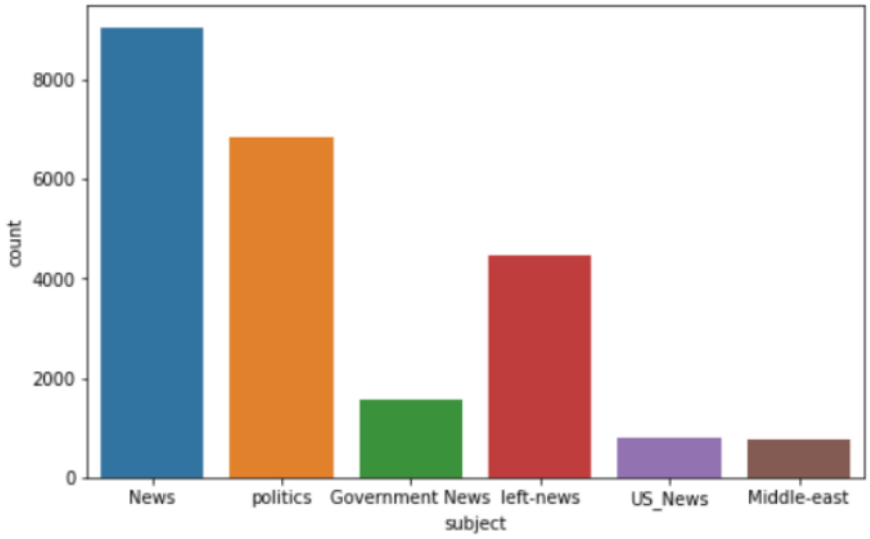
Counting by Subjects

```
for key,count in fake.subject.value_counts().iteritems():print(f"{key}:\t{count}")

#Getting Total Rows
print(f"Total Records:\t{fake.shape[0]}")
```

```
News:   9050
politics:       6841
left-news:      4459
Government News:        1570
US_News:        783
Middle-east:    778
Total Records:  23481
```

```
plt.figure(figsize=(8,5)) sns.countplot("subject",
data=fake)plt.show()
```

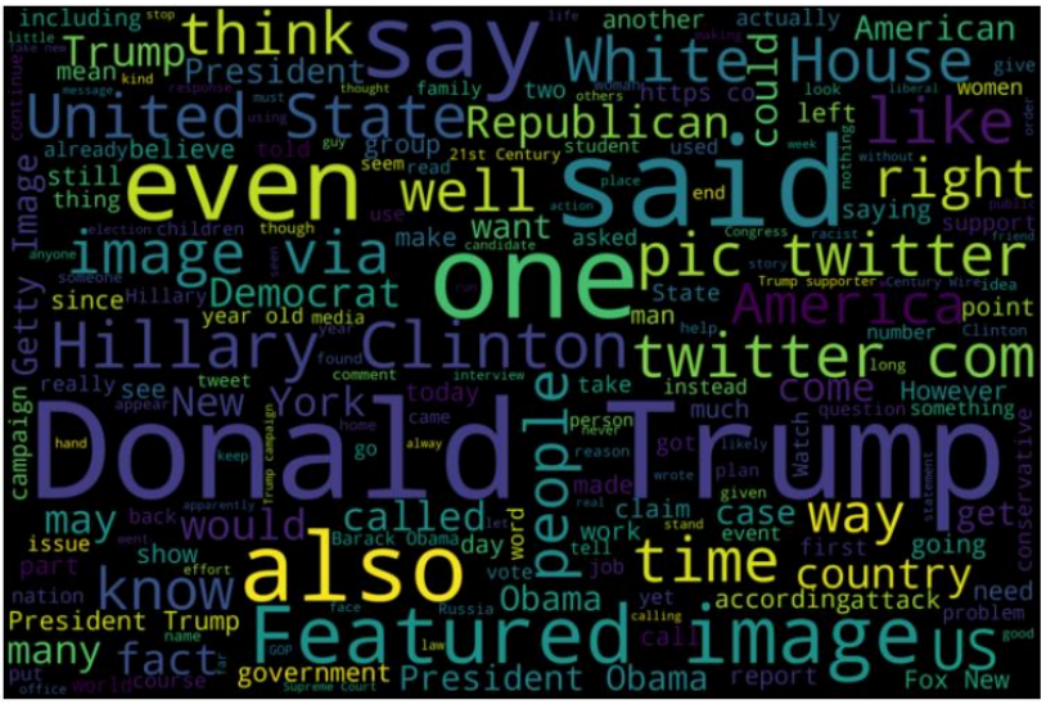


Word Cloud

```
text = ''
for news in fake.text.values:text += f" {news}"
wordcloud = WordCloud(width =
        3000,



        height = 2000, background_color = 'black',
        stopwords =
set(nltk.corpus.stopwords.words("english"))).generate(text)fig = plt.figure(
        figsize = (40, 30),facecolor = 'k',
        edgecolor = 'k')
plt.imshow(wordcloud, interpolation = 'bilinear')plt.axis('off')
plt.tight_layout(pad=0)plt.show()
del text
```



Exploring Real/ True News

```
real = pd.read_csv("/kaggle/input/fake-and-real-news-dataset/True.csv")
real.head()
```

	title	text	subject	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017

Difference in Text

Real news seems to have source of publication which is not present in fake news setLooking at the data:

- most of text contains reuters information such as "**WASHINGTON (Reuters)**".
- Some text are tweets from Twitter
- Few text do not contain any publication info

Cleaning Data

Removing Reuters or Twitter Tweet information from the text

- Text can be splitted only once at " - " which is always present after mentioningsource of publication, this gives us publication part and text part
- If we do not get text part, this means publication details was't given for that record
- The Twitter tweets always have same source, a long text of max 259 characters

Creating list of index that do not have publication part

```
unknown_publishers = []
for index,row in enumerate(real.text.values):
    try:
        record = row.split(" -", maxsplit=1)
        #if no text part is present, following will give error
        record[1]
        #if len of piblication part is greater than 260
        #following will give error, ensuring no text having "-" inbetween is counted
        assert(len(record[0]) < 260)
    except:
        unknown_publishers.append(index)
```

List of indices where publisher is not mentioned

```
real.iloc[unknown_publishers].text
#true, they do not have text like "WASHINGTON (Reuters)"
```

```
3488     The White House on Wednesday disclosed a group...
4358     Neil Gorsuch, President Donald Trump's appoint...
4465     WASHINGTON The clock began running out this we...
5784     Federal appeals court judge Neil Gorsuch, the ...
6660     Republican members of Congress are complaining...
6823     Over the course of the U.S. presidential campa...
7922     After going through a week reminiscent of Napo...
8194     The following timeline charts the origin and s...
8195     Global health officials are racing to better u...
8247     U.S. President Barack Obama visited a street m...
8465     ALGONAC, MICH.—Parker Fox drifted out of the D...
8481     Global health officials are racing to better u...
8482     The following timeline charts the origin and s...
8505     Global health officials are racing to better u...
8506     The following timeline charts the origin and s...
8771     In a speech weighted with America's complicate...
8970
9008     The following timeline charts the origin and s...
9009     Global health officials are racing to better u...
9307     It's the near future, and North Korea's regime...
9618     GOP leaders have unleashed a stunning level of...
9737     Caitlyn Jenner posted a video on Wednesday (Ap...
10479    The Democratic and Republican nominees for the...
Name: text, dtype: object
```

While looking at texts that do not contain publication info such as which reuter, we noticedone thing.

Text at index 8970 is empty

real.iloc[8970]

```
title      Graphic: Supreme Court roundup
text
subject                      politicsNews
date                       June 16, 2016
Name: 8970, dtype: object
```

Seperating Publication info, from actual text

```
publisher = []tmp_text = []
for index,row in enumerate(real.text.values):


    if index in unknown_publishers:
        #Add unknown of publisher not mentioned
        tmp_text.append(row)

        publisher.append("Unknown")
        continue
    record = row.split(" -", maxsplit=1)
    publisher.append(record[0]) tmp_text.append(record[1])
```

Replace existing text column with new text

Add seperate column for publication info

```
real["publisher"] = publisherreal["text"] =
tmp_text

del publisher, tmp_text, record, unknown_publishers

real.head()
```

	title	text	subject	date	publisher
0	As U.S. budget fight looms, Republicans flip t...	The head of a conservative Republican faction...	politicsNews	December 31, 2017	WASHINGTON (Reuters)
1	U.S. military to accept transgender recruits o...	Transgender people will be allowed for the fi...	politicsNews	December 29, 2017	WASHINGTON (Reuters)
2	Senior U.S. Republican senator: 'Let Mr. Muell...	The special counsel investigation of links be...	politicsNews	December 31, 2017	WASHINGTON (Reuters)
3	FBI Russia probe helped by Australian diplomat...	Trump campaign adviser George Papadopoulos to...	politicsNews	December 30, 2017	WASHINGTON (Reuters)
4	Trump wants Postal Service to charge 'much mor...	President Donald Trump called on the U.S. Pos...	politicsNews	December 29, 2017	SEATTLE/WASHINGTON (Reuters)

New column called "Publisher" has been added.

4

Checking for rows with empty text like row:8970

```
[index for index,text in enumerate(real.text.values) if
str(text).strip() == '']

#dropping this record
real = real.drop(8970, axis=0)
```

Checking for the same in fake news

```
empty_fake_index = [index for index,text in enumerate(fake.text.values) if str(text).strip()
== '']print(f"No of empty rows: {len(empty_fake_index)}")
fake.iloc[empty_fake_index].tail()
```

	title	text	subject	date
21816	BALTIMORE BURNS: MARYLAND GOVERNOR BRINGS IN N...		left-news	Apr 27, 2015
21826	FULL VIDEO: THE BLOCKBUSTER INVESTIGATION INTO...		left-news	Apr 25, 2015
21827	(VIDEO) HILLARY CLINTON: RELIGIOUS BELIEFS MUS...		left-news	Apr 25, 2015
21857	(VIDEO)ICE PROTECTING OBAMA: WON'T RELEASE NAM...		left-news	Apr 14, 2015
21873	(VIDEO) HYSTERICAL SNL TAKE ON HILLARY'S ANNOU...		left-news	Apr 12, 2015

630 Rows in Fake news with empty text

I've also observed a lot of CPATIAL-CASES in bogus news. Could keep letter cases, howeverwe'll be utilising Google's pretrained word2vec vectors later on, which have well-formed lower case words. We will try to use lower case.

The text for these rows appears to be included in the title. Let's combine the title and textto solve these problems.

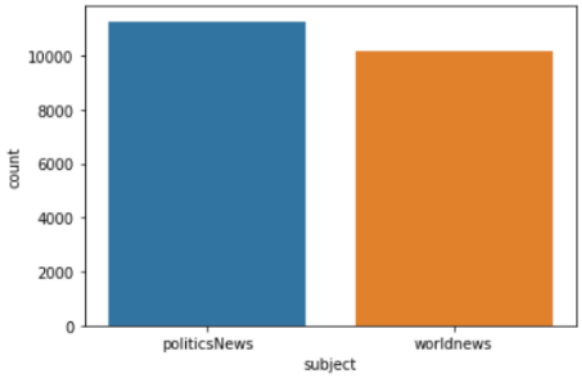
Getting Total Rows

```
print(f"Total Records:\t{real.shape[0]}")
```

```
Total Records:  21416
politicsNews:   11271
worldnews:      10145
```

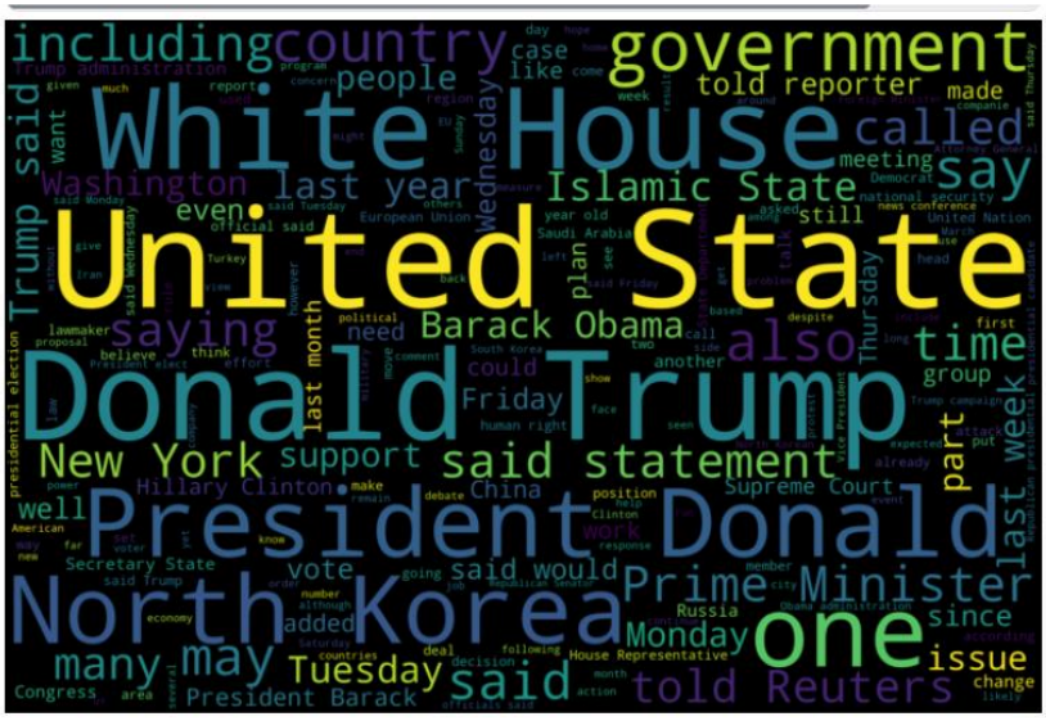
```
# Counting by Subjects
for key,count in real.subject.value_counts().iteritems():print(f"{key}:\t{count}")

sns.countplot(x="subject", data=real)plt.show()
```



WordCloud For Real News

```
text = ''
for news in real.text.values:text += f" {news}"
wordcloud = WordCloud(width =
        3000,
        height = 2000, background_color = 'black',
        stopwords =
set(nltk.corpus.stopwords.words("english"))).generate(str(text))fig = plt.figure(
        figsize = (40, 30),facecolor = 'k',
        edgecolor = 'k')
plt.imshow(wordcloud, interpolation = 'bilinear')plt.axis('off')
plt.tight_layout(pad=0)plt.show()
del text
```



Preprocessing Text Adding class Information

```
real["class"] = 1
fake["class"] = 0
```

Combining Title and Text

```
real["text"] = real["title"] + " " + real["text"]
fake["text"] = fake["title"] + " " + fake["text"]


Subject is Diffrent for real and fake => dropping
real = real.drop(["subject", "date","title", "publisher"], axis=1)fake = fake.drop(["subject", "date",
"title"], axis=1)


 Combining both into new dataframe
data = real.append(fake, ignore_index=True)


del real, fake

Removing StopWords, Punctuations and single-character words

 Converting X to format acceptable by gensim, removing anndpunctuation stopwords in the process
= data["class"].valuesX = []
stop_words = set(nltk.corpus.stopwords.words("english"))tokenizer =
nltk.tokenize.RegexpTokenizer(r'\w+')
for par in data["text"].values:tmp = []
        sentences = nltk.sent_tokenize(par)
        for sent in sentences: sent =
                sent.lower()
                tokens = tokenizer.tokenize(sent)
                filtered_words = [w.strip() for w in tokens if w not in
stop_words and len(w) > 1]
                tmp.extend(filtered_words)X.append(tmp)


del data

Vectorization -- Word2Vec

Word2Vec is one of the most popular technique to learn word embeddings using shallowneural network. It
was developed by Tomas Mikolov in 2013 at Google.

Word embedding is the most popular representation of document vocabulary. It is capableof capturing
context of a word in a document, semantic and syntactic similarity, relation with other words, etc.

Let's create and check our own Word2Vec model with gensim
import gensim


 Dimension of vectors we are generating
EMBEDDING_DIM = 100

#Creating Word Vectors by Word2Vec Method
```

```
w2v_model = gensim.models.Word2Vec(sentences=X, size=EMBEDDING_DIM,window=5, min_count=1)
```

Vocab Size

```
len(w2v_model.wv.vocab)
```

 # Represented each of 122248 words by a 100dim vector.

Exploring Vectors

8

See a sample vector for random word, say Corona

```
w2v_model["corona"]
```

```
array([-0.05927228, -0.01596732,  0.00595484,  0.0357627 , -0.01360779,
       -0.00170421, -0.05227572, -0.02337155, -0.06498439,  0.02310319,
       -0.08135039, -0.06741004,  0.01060605, -0.00233191,  0.04859036,
       -0.00193688,  0.00489498, -0.01469864, -0.05440938,  0.05713286,
        0.04011549,  0.0378617 , -0.06951197, -0.00642549,  0.03158214,
        0.00401993, -0.01131374, -0.0488236 ,  0.01623606, -0.00864428,
        0.02174502,  0.02032683, -0.02228736,  0.05060292, -0.00439747,
       -0.04379179, -0.078948  , -0.00844735,  0.01823636,  0.05118509,
       -0.03192174,  0.02387427, -0.01070322, -0.02082633,  0.01906741,
       -0.03466871, -0.01808968,  0.00522737, -0.02900375, -0.00436146,
       -0.01538342,  0.05872579, -0.00945784, -0.02772546, -0.04476467,
        0.03220233, -0.01844146, -0.07554039, -0.0406533 , -0.00573117,
       -0.05629358, -0.01685986, -0.04480005,  0.01407769, -0.03635843,
       -0.02661304, -0.00449126,  0.06267284,  0.08125523,  0.00871364,
        0.04246728, -0.01293021,  0.07562466,  0.0441062 ,  0.07445565,
        0.05795208, -0.03853748, -0.04694097,  0.03917553, -0.02983577,
        0.01450436, -0.03755805, -0.02396097, -0.01661294,  0.0267911 ,
       -0.00579214, -0.00870581, -0.01720314,  0.01761026, -0.00528431,
       -0.03443367,  0.02185205, -0.04053083, -0.01236498,  0.01519614,
        0.01834148,  0.04276792,  0.00768779, -0.02659689, -0.03393488],
      dtype=float32)
```

7

```
w2v_model.wv.most_similar("iran")
```

```
[('tehran', 0.8996585607528687),
 ('iranian', 0.7465806007385254),
 ('destabilizing', 0.6763759851455688),
 ('nuclear', 0.6622727513313293),
 ('iranians', 0.6291905641555786),
 ('hezbollah', 0.6274085640907288),
 ('riyadh', 0.626042366027832),
 ('qatar', 0.6157292127609253),
 ('pyongyang', 0.6120718717575073),
 ('jcpoa', 0.6091011762619019)]
```

```
w2v_model.wv.most_similar("fbi")
```

```
[('comey', 0.7295312285423279),
 ('cia', 0.6076300144195557),
 ('investigators', 0.6040865182876587),
 ('investigation', 0.5735664367675781),
 ('mueller', 0.5696539878845215),
 ('doj', 0.5647487640380859),
 ('whosonfirst', 0.5577248930931091),
 ('investigations', 0.5507204532623291),
 ('inquiry', 0.5251892805099487),
 ('probe', 0.5228870511054993)]
```

w2v_model.wv.most_similar("facebook")

```
[('fb', 0.6758648157119751),
 ('reddit', 0.6643215417861938),
 ('google', 0.6571981310844421),
 ('gofundme', 0.6353918313980103),
 ('instagram', 0.6272882223129272),
 ('online', 0.575968861579895),
 ('blog', 0.5716950297355652),
 ('website', 0.5683227777481079),
 ('posting', 0.5662258863449097),
 ('614383415410204', 0.5657731294631958)]
```

w2v_model.wv.most_similar("computer")

```
[('computers', 0.8432672023773193),
 ('software', 0.7982585430145264),
 ('laptop', 0.7835859060287476),
 ('malware', 0.770531177520752),
 ('electronic', 0.7449864745140076),
 ('kaspersky', 0.7325843572616577),
 ('servers', 0.7234582901000977),
 ('scanning', 0.7193918228149414),
 ('devices', 0.6898456811904907),
 ('stored', 0.689788818359375)]
```

8 **Feeding US Presidents**

w2v_model.wv.most_similar(positive=["trump","obama", "clinton"])

#First was Bush

```
[('elect', 0.5307915210723877),
 ('bush', 0.5218258500099182),
 ('cruz', 0.5085068941116333),
 ('incoming', 0.4915129840373993),
 ('course', 0.4905095100402832),
 ('bartlet', 0.4892512857913971),
 ('americas_dad', 0.48754456639289856),
 ('actually', 0.4863002896308899),
 ('crooked', 0.47504839301109314),
 ('outright', 0.46710726618766785)]
```

Looking at the similar words, vectors are well formed for these words :)

These Vectors will be passed to LSTM/GRU instead of words. 1D-CNN can further be usedto extract features from the vectors.

Keras has implementation called "**Embedding Layer**" which would create word embeddings(vectors). Since we did that with gensim's word2vec, we will load these vectorsinto embedding layer and make the layer non-trainable.

Tokenizer can represent each word by number. Since we cannot pass string words toembedding layer, thus need some way to represent each words by numbers.

Tokenizing Text -> Repsesenting each word by a number

Mapping of orginal word to number is preserved in word_indexproperty of tokenizer

Tokenized applies basic processing like changing it yo lower case,explicitely setting that as False

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X)

X = tokenizer.texts_to_sequences(X)
```

Checking the first 10 words of first news # every word has beenrepresented with a number

```
X[0][:10]
```

Mapping is preserved in dictionary -> word_index property of instance

```
word_index = tokenizer.word_index
for word, num in word_index.items():print(f"{word} ->
    {num}")
        if num == 10:
            break
```

```
trump -> 1
said -> 2
president -> 3
would -> 4
people -> 5
one -> 6
state -> 7
new -> 8
obama -> 9
also -> 10
```

Notice it starts with 1

We can pass numerical representation of words into neural network.

We can use Many-To-One (Sequence-To-Word) Model of RNN, as we have many words innews as input and one output ie Probability of being Real.

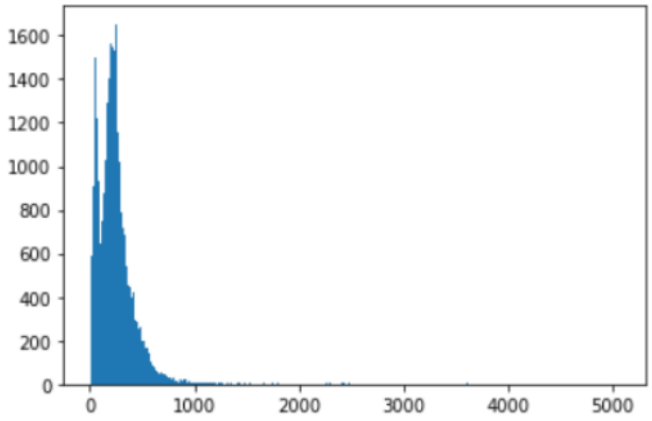
For Many-To-One model, lets use a fixed size input.

Histogram for no of words in news shows that most news article areunder 700 words.

Keeping each news small and truncate all news to 700 whiletokenizing

```
plt.hist([len(x) for x in X], bins=500)
plt.show()
```

```
# Its heavily skewed.
# Truncate these outliers
```




```
no = np.array([len(x) for x in X])len(nos[nos < 700])
# Out of 48k news, 44k have less than 700 words
```

Keep all news to 700, add padding to news with less than 700 wordsand truncating long ones

```
maxlen = 700

#Making all news of size maxlen defined above
X = pad_sequences(X, maxlen=maxlen)
#all news has 700 words (in numerical form now). If they had less words, they have beenpadded with 0
```

```
# 0 is not associated to any word, as mapping of words started from 1# 0 will also be used later, if
unknows word is encountered in test set
len(X[0])
# Adding 1 because of reserved 0 index
# Embedding Layer creates one more vector for "UNKNOWN" words, orpadded words (0s). This
Vector is filled with zeros.
# Thus our vocab size inceeases by 1
vocab_size = len(tokenizer.word_index) + 1
```

Function to create weight matrix from word2vec gensim model

```
def get_weight_matrix(model, vocab):
    # total vocabulary size plus 0 for unknown words
    vocab_size = len(vocab) + 1
    # define weight matrix dimensions with all 0
    weight_matrix = np.zeros((vocab_size, EMBEDDING_DIM))



    # step vocab, store vectors using the Tokenizer's integer mapping
    for word, i in vocab.items(): weight_matrix[i] =
        model[word]
    return weight_matrix
```

We Create a matrix of mapping between word-index and vectors. We use this as weights inembedding layer

Embedding layer accepts numecical-token of word and outputs corresponding vercor toinner layer.

It sends vector of zeros to next layer for unknown words which would be tokenized to 0.

Input length of Embedding Layer is the length of each news (700 now due to padding andtruncating)

Getting embedding vectors from word2vec and usings it as weights of non-trainable keras embedding layer

```
embedding_vectors = get_weight_matrix(w2v_model, word_index)
```

Defining Neural Network

```
model = Sequential()
#Non-trainable embeddidng layer
model.add(Embedding(vocab_size, output_dim=EMBEDDING_DIM,
weights=[embedding_vectors], input_length=maxlen, trainable=False))#LSTM
model.add(LSTM(units=128)) model.add(Dense(1,
activation='sigmoid'))
```

```
model.compile(optimizer='adam', loss='binary_crossentropy',metrics=['acc'])

del embedding_vectors

model.summary() #Train
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 700, 100)          12224900
_________________________________________________________________
lstm (LSTM)                  (None, 128)               117248
_________________________________________________________________
dense (Dense)                (None, 1)                 129
=================================================================
Total params: 12,342,277
Trainable params: 117,377
Non-trainable params: 12,224,900
_________________________________________________________________
```

test split

```
X_train, X_test, y_train, y_test = train_test_split(X, y)

model.fit(X_train, y_train, validation_split=0.3, epochs=6)
```

```
Train on 23570 samples, validate on 10102 samples
Epoch 1/6
23570/23570 [==============================] - 37s 2ms/sample - loss: 0.1267 - ac
c: 0.9541 - val_loss: 0.1607 - val_acc: 0.9264
Epoch 2/6
23570/23570 [==============================] - 32s 1ms/sample - loss: 0.0691 - ac
c: 0.9765 - val_loss: 0.0632 - val_acc: 0.9788
Epoch 3/6
23570/23570 [==============================] - 32s 1ms/sample - loss: 0.0445 - ac
c: 0.9852 - val_loss: 0.0537 - val_acc: 0.9834
Epoch 4/6
23570/23570 [==============================] - 32s 1ms/sample - loss: 0.0327 - ac
c: 0.9890 - val_loss: 0.0649 - val_acc: 0.9786
Epoch 5/6
23570/23570 [==============================] - 32s 1ms/sample - loss: 0.0210 - ac
c: 0.9934 - val_loss: 0.0349 - val_acc: 0.9885
Epoch 6/6
23570/23570 [==============================] - 32s 1ms/sample - loss: 0.0220 - ac
c: 0.9921 - val_loss: 0.0782 - val_acc: 0.9775
```

Prediction is in probability of news being real, so converting intoclasses

```
# Class 0 (Fake) if predicted prob < 0.5, else class 1 (Real)
y_pred = (model.predict(X_test) >= 0.5).astype("int")

accuracy_score(y_test, y_pred)
```

```
: 0.9802227171492205
```

```
print(classification_report(y_test, y_pred))
```

```
                         precision    recall  f1-score   support

                      0       0.97      1.00      0.98      5867
                      1       1.00      0.96      0.98      5358

               accuracy                           0.98     11225
              macro avg       0.98      0.98      0.98     11225
           weighted avg       0.98      0.98      0.98     11225
```

```
del model
```

Using Pre-Trained Word2Vec Vectors

Requirements RAM: 12GB and HardDisk Space: 4GB

Invoke garbage collector to free ram

```
import gc gc.collect()

from gensim.models.keyedvectors import KeyedVectors

# Requires RAM
word_vectors = KeyedVectors.load_word2vec_format('../input/googlenewsvectorsnegative3
00/GoogleNews-vectors-negative300.bin',    binary=True) EMBEDDING_DIM=300
```

Exploring these trained Vectors

```
embedding_matrix = np.zeros((vocab_size, EMBEDDING_DIM))
for word, i in word_index.items():
        try:
#                 embedding_vector = word_vectors[word]
                embedding_matrix[i] = embedding_vector
        except KeyError: embedding_matrix[i]=np.random.normal(0,np.sqrt(0.25),EMBEDDING_DIM)

# del word_vectors

model = Sequential()
model.add(Embedding(vocab_size, output_dim=EMBEDDING_DIM, weights=[embedding_matrix],
input_length=maxlen, trainable=False))model.add(Conv1D(activation='relu', filters=4, kernel_size=4))
model.add(MaxPool1D())
model.add(LSTM(units=128)) model.add(Dense(1,
activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy',metrics=['acc'])

del embedding_matrix

model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 700, 300)          36674700
_________________________________________________________________
conv1d (Conv1D)              (None, 697, 4)            4804
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 348, 4)            0
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               68096
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 36,747,729
Trainable params: 73,029
Non-trainable params: 36,674,700
_________________________________________________________________
```

model.fit(X_train, y_train, validation_split=0.3, epochs=12)

```
Train on 23570 samples, validate on 10102 samples
Epoch 1/12
23570/23570 [==============================] - 25s 1ms/sample - loss: 0.3121 - ac
c: 0.8832 - val_loss: 0.2150 - val_acc: 0.9249
Epoch 2/12
23570/23570 [==============================] - 23s 968us/sample - loss: 0.2342 - a
cc: 0.9109 - val_loss: 0.2138 - val_acc: 0.9210
Epoch 3/12
23570/23570 [==============================] - 23s 987us/sample - loss: 0.1996 - a
cc: 0.9257 - val_loss: 0.1748 - val_acc: 0.9379
Epoch 4/12
23570/23570 [==============================] - 23s 997us/sample - loss: 0.1936 - a
cc: 0.9267 - val_loss: 0.1415 - val_acc: 0.9555
Epoch 5/12
23570/23570 [==============================] - 23s 991us/sample - loss: 0.1911 - a
cc: 0.9274 - val_loss: 0.1379 - val_acc: 0.9490
Epoch 6/12
23570/23570 [==============================] - 23s 990us/sample - loss: 0.0974 - a
cc: 0.9648 - val_loss: 0.0852 - val_acc: 0.9677
Epoch 7/12
23570/23570 [==============================] - 24s 998us/sample - loss: 0.1610 - a
cc: 0.9399 - val_loss: 0.1060 - val_acc: 0.9648
Epoch 8/12
23570/23570 [==============================] - 23s 989us/sample - loss: 0.0892 - a
cc: 0.9681 - val_loss: 0.1001 - val_acc: 0.9642
Epoch 9/12
23570/23570 [==============================] - 23s 983us/sample - loss: 0.0712 - a
cc: 0.9761 - val_loss: 0.0812 - val_acc: 0.9710
Epoch 10/12
23570/23570 [==============================] - 24s 1ms/sample - loss: 0.0546 - ac
c: 0.9816 - val_loss: 0.0798 - val_acc: 0.9705
Epoch 11/12
23570/23570 [==============================] - 23s 992us/sample - loss: 0.0472 - a
cc: 0.9842 - val_loss: 0.0682 - val_acc: 0.9757
Epoch 12/12
23570/23570 [==============================] - 23s 982us/sample - loss: 0.0416 - a
```

```
y_pred = (model.predict(X_test) > 0.5).astype("int")

accuracy_score(y_test, y_pred)
```

```
0.9813808463251671
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      5867
           1       0.98      0.98      0.98      5358

    accuracy                           0.98     11225
   macro avg       0.98      0.98      0.98     11225
weighted avg       0.98      0.98      0.98     11225
```

ORIGINALITY REPORT

36%	32%	7%	29%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Westminster Student Paper	17%
2	www.kaggle.com Internet Source	9%
3	www.codercto.com Internet Source	4%
4	Submitted to Sheffield Hallam University Student Paper	1%
5	archive.org Internet Source	1%
6	dokumen.pub Internet Source	1%
7	Submitted to University of East London Student Paper	1%
8	Submitted to Infile Student Paper	<1%
9	Chetan Agrawal, Anjana Pandey, Sachin Goyal. "Fake news detection system based on modified bi-directional long short term	<1%

memory", Multimedia Tools and Applications, 2022
Publication

10	Mohamed K. Elhadad, Kin Fun Li, Fayez Gebali. "Chapter 86 A Novel Approach for Selecting Hybrid Features from Online News Textual Metadata for Fake News Detection", Springer Science and Business Media LLC, 2020 Publication	<1 %
11	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
12	www.safetylit.org Internet Source	<1 %
13	Muhammad Shahid Anwar, Jing Wang, Sadique Ahmad, Wahab Khan, Asad Ullah, Mudassir Shah, Zesong Fei. "Impact of the Impairment in 360-degree Videos on Users VR Involvement and Machine Learning-Based QoE Predictions", IEEE Access, 2020 Publication	<1 %
14	"Disinformation, Misinformation, and Fake News in Social Media", Springer Science and Business Media LLC, 2020 Publication	<1 %
15	Submitted to CSU, San Jose State University Student Paper	<1 %

16	"Combating Fake News with Computational Intelligence Techniques", Springer Science and Business Media LLC, 2022 Publication	<1 %
17	"Data Engineering for Smart Systems", Springer Science and Business Media LLC, 2022 Publication	<1 %
18	Submitted to International Hellenic University Student Paper	<1 %
19	doctorpenguin.com Internet Source	<1 %
20	guptaanshul.me Internet Source	<1 %
21	praison.com Internet Source	<1 %
22	Ameyaa Biwalkar, Ashwini Rao, Ketan Shah. "Real or Fake: An intrinsic analysis using supervised machine learning algorithms", 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2021 Publication	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

9

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34