Fake news detection using NLP.

A Project Report in partial fulfillment of the degree

Bachelor of Technology

in

Electronics & Communication Engineering/Computer Science & Engineering

By

19K41A04A6	D. Meghana
19K41A0563	Anju Haliya
19K41A0565	B. Sreshta

Under the Guidance of
D. Ramesh

Submitted to

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
S R ENGINEERING COLLEGE(A), ANANTHASAGAR, WARANGAL
(Affiliated to JNTUH, Accredited by NBA)

Dec-2022

SR Engineering College
Innovation . Creativity . Entrepreneurship

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled "Fake news detection using NLP" is a record of bonafide work carried out by the student(s) D. Meghana, Anju Haliya, B. Srehta bearing Roll No(s) 19K41A04A6, 19K41A0563, 19K41A0565 during the academic year 2022-2023 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Electronics & Communication/Computer Science Engineering** by the Jawaharlal Nehru Technological University, Hyderabad.

.

Supervisor **Head of the Department**

External Examiner

ABSTRACT

The idea of access to authentic news is the foundation of our project. As internet as evolved to be a dominant reliance for everything and social media has become to be the news providing factor for many. There is rigorous spread of unauthentic or fake news to these chances which are inhibiting the threat to the democracy. The fake news spreading is taken up for much wrong intentions of defamation, back stabbing the legit, biased support for a candidate in election etc. These circumstance would result in threat for an innocent and lack of essential awareness and proper decision making in audience. Moreover, these spread news are made to encash through advertisements by entitling attention grabbing headlines to the click-baits. Therefore, we have understood the necessity to restore the public trust and access to real and genuine news to them. We have referred concepts of NLP - Natural Language Processing for the same and extracted knowledge through our literature survey. We have interfaced the concepts of word2vec and LSTM to carryout word embedding and text classification. To sum up, our model works at a good accuracy of 95 and helps aid the readers with the comfort of the authentic trustworthy news to the best possible.

Table of Contents

1. INTRODUCTION:

The main goal of our proposed system is to identify the fake news and ensure the benefits of the authentic news to sector of audience because they have a cumulative impact on several aspects of life, our regulations and democracy. Internet has made world a small place to connect collaborate, know things and learn ideologies thereby depending on the same for the news too. Nevertheless, there is a credible authenticity we should be grateful for this technology to smarten our lives in providing news with ease of access.But this has been taken up as an advantage to misuse the resource by spammers, cyber experts, influence-rs and wicked persons. Alarmingly, the concept of robots being employed to fabricate news and spread lies should be a topic of concern. All these activities are taken forward for various discriminative benefits of an unapplaudable political agenda, diversified notions, monetizing through clicks, saving the accused etc. Therefore, we have realized the necessity to distinguish the fake news from the real news and advance our research accordingly. Then we understood the word embedding concept and used word2vec model for the same. The embedded text is classified using the neural structure of LSTM - Long short term memory because this model is the best for sequential classification.

2. LITERATURE REVIEW

In the various research papers we have referred that different Machine learning Algorithms have been used. The area of Artificial intelligence has been the suitable criteria to carry out predictions on the datasets by feature extraction and data pre-processing. The various machine learning algorithms that have been used are : Logistic regression, Support Vector Machine, Linear Regression, Discriminant analysis stocastic gradient descent and ridge regression, Naive Bayes classification, Deep learning, TF-IDF Vectorization, LSTM, hash vectorizer, Random Forest, Decision tree, Adaboost and rule based classifier.

Having an insight into all these algorithms, we have observed that the algoithms work different by generating the pattern among the available dataset and proceeding with predition. The concepts of logistic regression, ridge regression, stocastic gradient, SVM are used wth algorithms especialaly which follows a close correlation among the variables taken into consideration whereas decision tree, random forest, adaboost works with regenerating similarities by nodes, Deep learning is something that works by generating biases and weights in the layers, rule based takes the bulk values and signifies a rule in it, LSTM takes into account the input, output and forget gates considering the memories of occurances of the informaion.

Of all when sentences and language is a mattern of concern, NLP - Natural LanguageProcessing is the best methodology bcause they have those kind of algorithms that infer knowledge and interpretations of the txt and voice data just as we humans do. The packages of NTP are best suitable to proceed with our project and the dataset. In NLP the fed information is first converted into vector form and those numerical figures manipulate thorough the cycles and facilitates interpretations and clssifications. The recommonded algoritm to do the word embedding for data preprocessing is word2vec.

SI NO	DATE OF PUBLICATION	AUTHORS	NAME	METHADOLOGY	DATASET	ACCURACY
1	29 July, 2021.	Arvin Hansrajh Timothy T. Adeliyi Jeanette Wing1	Detection of Online Fake News Using Blending Ensemble Learning	logistic regression support vector machine linear discriminant analysis stochastic gradient descent and ridge regression	LIAR ISOT	60.8 98.4
2	29 November, 2018.	Akshay Jain Amey Kasbe	Fake News Detection	Naive Bayes classification	GIT HUB - Labelled as authentic.	Title : 80 Text : 93
3	April, 2021.	Jamal AbdulNasirOsama SubhaniKhan IraklisVarlamis	Fake news detection: A hybrid CNN-RNN based deep learning approach	Deep learning : CNN - RNN	FA-KES ISOT	60 99
4	June, 2022.	Hemalatha A Karpahalakshmi S Thanga Sri R Vaishnavi M Bhavani N	Fake News Detection Using Machine Learning	Feature extraction vectorization : Count Vectorizer and Tiff Vectorizer From Python scikit-learn library SVM Naive Bayes	fake and real news from the online media.	98.6
5	April, 2021.	Tejaswini Yesugade Shrikant Kokate Sarjana Patil Ritik Varma Sejal Pawar	Fake News Detection Using LSTM	LSTM	From Kaggle	91.5
6	July, 2021.	kajal Kumari	Detecting Fake News with Natural Language Processing	TfidfVectorizer Count Vectorizer Hash Vectorizer	From Kaggle	93.2
7	May, 2021.	Nabanita Roy	Predicting Fake News using NLP and Machine Learning/Scikit-learn/GloVe/Keras/ LSPM	Tf-idf and count vectorizer Using logistic regression Random Forest Decision Tree Gradient and Adaboost	Kaggle fake news dataset	96.5
8	October, 2021.	Barbara Probierz Piotr Stefański Jan Kozak	Rapid detection of Fake News based on Machine Learning Method	TfidfVectorizer Random Forest SVM CART Adaboost Bagging	Kaggle	99.6
9	August, 2021.	Phayung Meesad	Thai Fake News Detection Based on Information Retrieval, Natural Language Processing and Machine Learning	Naïve Bayesian Logistic Regression K-Nearest Neighbor Multilayer Perceptron Support Vector Machine Decision Tree Random Forest Rule-Based Classifier LSPM	Real, Fake and Suspicious	90.0
10	June, 2020.	Joyce Annie George	Fake News Detection Using NLP Techniques	SVM Count Vectorizer TfidfVectorizer	Fake and real news from Kaggle	99.6

3. DESIGN:

3.1 Requirement Specifications (S/W & H/W)

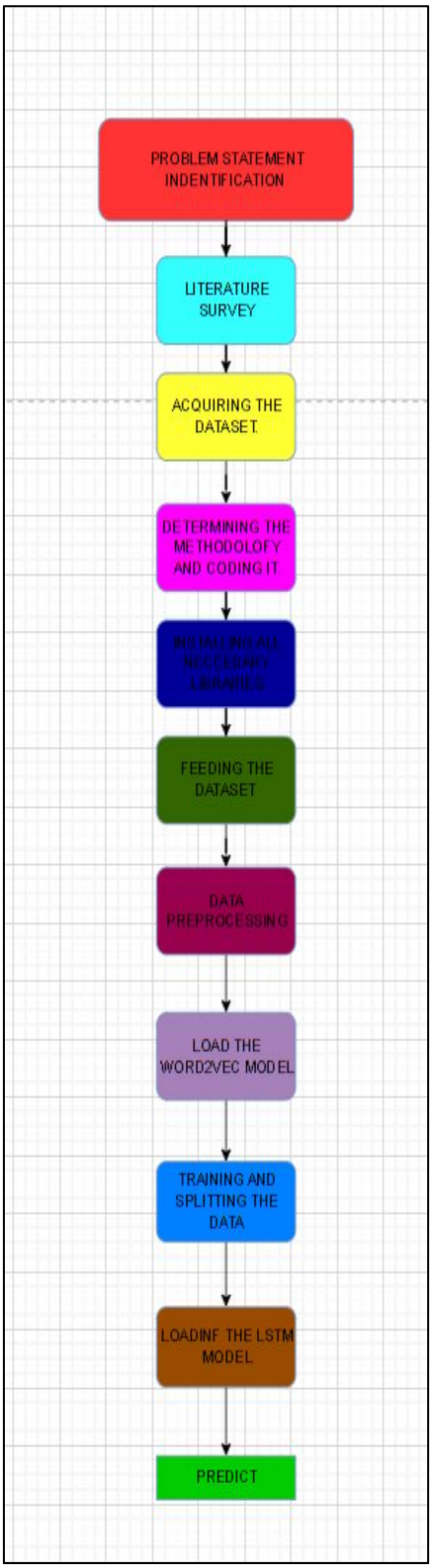
Hardware Requirements

- ✓ **System** : Processor Intel(R) Core (TM) i5-8265U CPU @ 1.60GHz, 1800 MHz, 4 Cores, 8 Logical Processors
- ✓ **RAM** : 8 GB
- ✓ **Hard Disk** : 557 GB
- ✓ **Input** : Keyboard and Mouse
- ✓ **Output** : PC

Software Requirements

- ✓ **OS** : Windows 10
- ✓ **Platform** : Google Colaboratory / Jupyter Notebook
- ✓ **Program Language** : Python

3.2 FLOW CHART



4. DATASET:

Dataset has been acquired from kaggle :

The thus collected news has features of :

Input features :

>id

> title

>author

>text

>Label (0 for ture 1 for fake)

Output feature :

> real or fake

dataset from kaggle of 20799 samples of fake and true text

```
[7]  df
```

	text	label
0	House Dem Aide: We Didn’t Even See Comey’s Let...	1
1	Ever get the feeling your life circles the rou...	0
2	Why the Truth Might Get You Fired October 29, ...	1
3	Videos 15 Civilians Killed In Single US Airstr...	1
4	Print \nAn Iranian woman has been sentenced to...	1
...	...	...
20795	Rapper T. I. unloaded on black celebrities who...	0
20796	When the Green Bay Packers lost to the Washing...	0
20797	The Macy’s of today grew from the union of sev...	0
20798	NATO, Russia To Hold Parallel Exercises In Bal...	1
20799	David Swanson is an author, activist, journa...	1

20800 rows × 2 columns

5. DATA PREPROCESSING:

Data pre-processing is essential while working on large dataset because algorithms could only be applied on the vectorized text. Data pre - processing thereby aims at covertin text into vectorized simple form which means tokenizing. Tokenizing means dividing the text into units of words or sentences. Tokenizing is the fundamental step for stemming and lemmatizaion.

We have elimated stop wards from the dataset as they have no significance in deciding the meaning of the text. Stemming has been applied to correlate the words belonging to same root.

Then we would proceed with label encoding, the label encoding is to signify the categorical data for the semi-structured or unstructured data. The label encoding means giving the labels for the data in numericals.

Next, stemming is done to produce morphological variants of a root/base word. Stemming programs are commonly referred to as stemming algorithms or stemmers. These algorithms are used to give the domain vocabularies in domain analysis.

Neural networks requires to have inputs with the same size. Therefore sentence inputs are padded with 0's after defining the max length and words are dropped and added accordingly.

6. METHODOLOGY:

After Data pre-processing we are going to perform word embedding using the Word2Vec vectorizer.

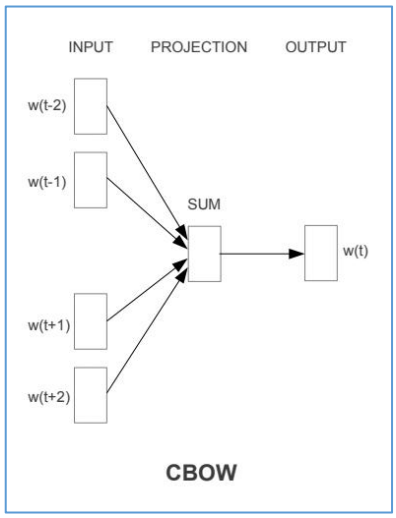
6.1 Word2Vec

Word2vec is not a singular algorithm, rather, it is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets. Embeddings learned through word2vec have proven to be successful on a variety of downstream natural language processing tasks.

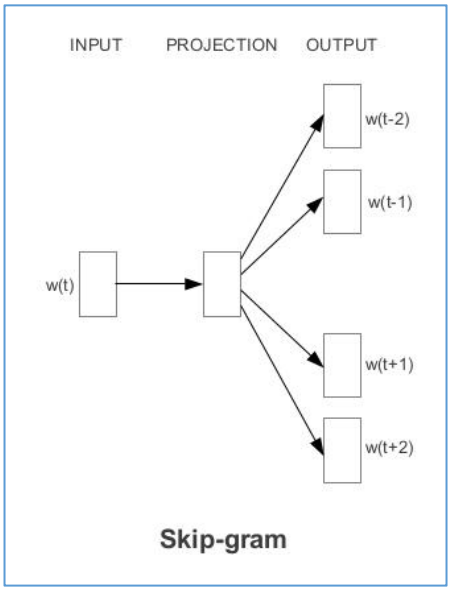
These papers proposed two methods for learning representations of words:

- **Continuous bag-of-words model**: predicts the middle word based on surrounding context words. The context consists of a few words before and after the current (middle) word. This

architecture is called a bag-of-words model as the order of words in the context is not important.



- **Continuous skip-gram model**: predicts words within a certain range before and after the current word. A worked example of this is given below.



After word embedding, we are going to load the LSTM model for text classification.

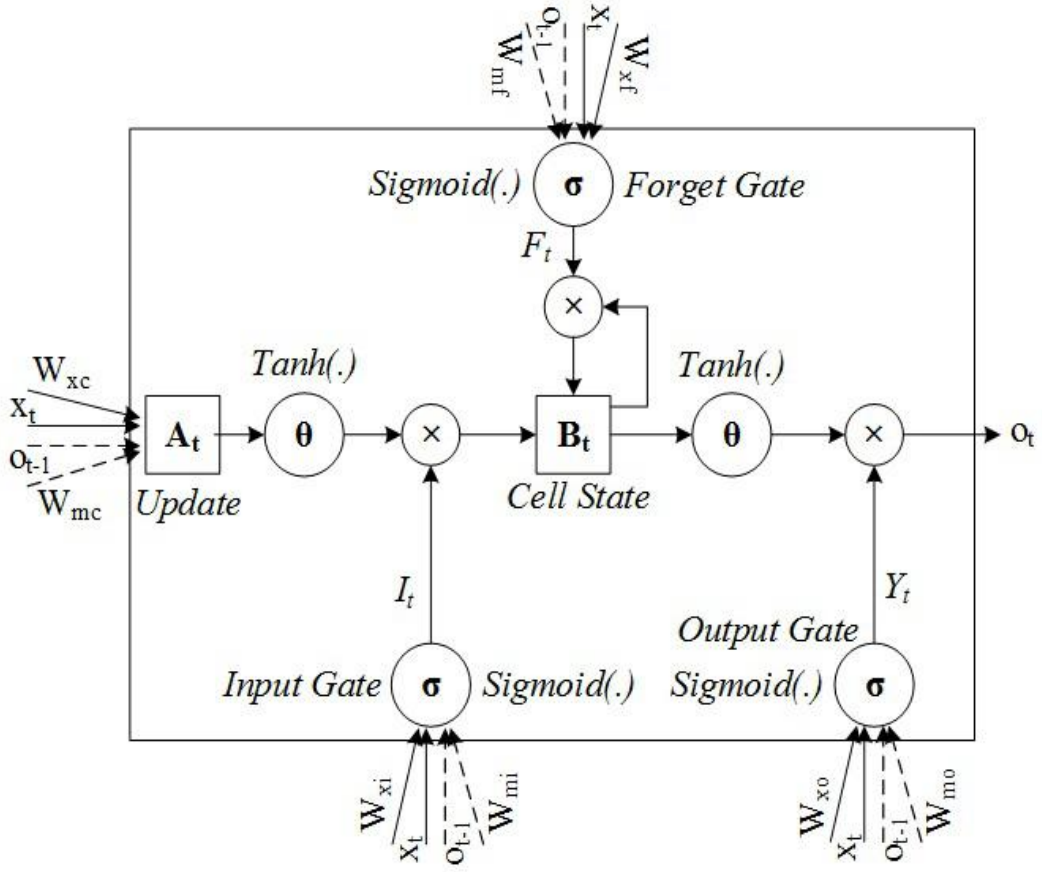
6.2 LSTM

LSTM (Long Short-Term Memory) network is a type of RNN (Recurrent Neural Network) that is widely used for learning sequential data prediction problems. As every other neural network LSTM also has some layers which help it to learn and recognize the pattern for better performance. The basic operation of LSTM can be considered to hold the required information and discard the information which is not required or useful for further prediction.

The Architecture of LSTM

A simple LSTM network consists of the following components.

- Forget gate
- Input gate.
- Output gate



7. RESULTS:

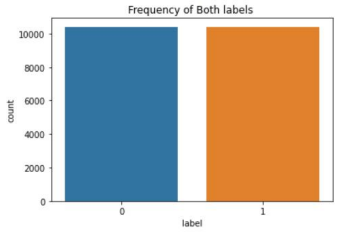
Our project gave out the accuracy of 95%

The output labels are the predictions if the data sample is real or not.

- The dataset containing samples of real and fake news.

```
[14] import seaborn as sns
     import matplotlib.pyplot as plt
     sns.countplot(df.label)
     plt.title('Frequency of Both labels ')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument
  FutureWarning
Text(0.5, 1.0, 'Frequency of Both labels ')
```




```
[15] X=df['text']
     Y=df.label
     X,Y
```

```
(0        House Dem Aide: We Didn’t Even See Comey’s Let...
 1        Ever get the feeling your life circles the rou...
 2        Why the Truth Might Get You Fired October 29, ...
 3        Videos 15 Civilians Killed In Single US Airstr...
 4        Print \nAn Iranian woman has been sentenced to...
                                ...
 20795    Rapper T. I. unloaded on black celebrities who...
 20796    When the Green Bay Packers lost to the Washing...
 20797    The Macy’s of today grew from the union of sev...
 20798    NATO, Russia To Hold Parallel Exercises In Bal...
 20799      David Swanson is an author, activist, journa...
 Name: text, Length: 20800, dtype: object, 0        1
 1        0
 2        1
 3        1
 4        1
         ..
 20795    0
 20796    0
 20797    0
 20798    1
 20799    1
 Name: label, Length: 20800, dtype: int64)
```

- Word embedding results

```
[34] len(x_train),len(ytrain),len(x_test),len(yval)
     embedding_matrix = np.zeros((vocab_size, 300))
     print(embedding_matrix)
     for word, i in tokenizer.word_index.items():
         if word in w2v_model.wv:
             embedding_matrix[i] = w2v_model.wv[word]
     print(embedding_matrix.shape)
```

```
[[0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]]
(133580, 300)
```

```
[35] embedding_matrix
```

```
array([[ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.47187069,  0.87185282, -0.90331572, ..., -0.67393523,
        -0.01811692, -0.24872576],
       [-1.23611009,  1.36165297,  0.324781  , ...,  0.69293916,
         0.1527236 , -1.94492328],
       ...,
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ]])
```

- LSTM model results

```
[39] from keras.callbacks import ReduceLROnPlateau, EarlyStopping
     callbacks = [ ReduceLROnPlateau(monitor='val_loss', patience=5, cooldown=0),EarlyStopping(monitor='val_acc', min_delta=1e-4, patience=5)]

     history = model.fit(x_train, ytrain,batch_size=32,epochs=10,validation_split=0.1,verbose=1,callbacks=callbacks)
```

```
Epoch 1/10
468/468 [==============================] - ETA: 0s - loss: 0.1382 - accuracy: 0.9444WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 173s 370ms/step - loss: 0.1382 - accuracy: 0.9444 - val_loss: 0.1571 - val_accuracy: 0.9411 - lr: 0.0010
Epoch 2/10
468/468 [==============================] - ETA: 0s - loss: 0.1223 - accuracy: 0.9518WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 173s 370ms/step - loss: 0.1223 - accuracy: 0.9518 - val_loss: 0.1657 - val_accuracy: 0.9411 - lr: 0.0010
Epoch 3/10
468/468 [==============================] - ETA: 0s - loss: 0.1054 - accuracy: 0.9585WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 173s 369ms/step - loss: 0.1054 - accuracy: 0.9585 - val_loss: 0.1635 - val_accuracy: 0.9399 - lr: 0.0010
Epoch 4/10
468/468 [==============================] - ETA: 0s - loss: 0.1037 - accuracy: 0.9595WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 174s 371ms/step - loss: 0.1037 - accuracy: 0.9595 - val_loss: 0.1632 - val_accuracy: 0.9423 - lr: 0.0010
Epoch 5/10
468/468 [==============================] - ETA: 0s - loss: 0.0913 - accuracy: 0.9655WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 175s 373ms/step - loss: 0.0913 - accuracy: 0.9655 - val_loss: 0.2508 - val_accuracy: 0.9363 - lr: 0.0010
Epoch 6/10
468/468 [==============================] - ETA: 0s - loss: 0.0837 - accuracy: 0.9679WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 174s 372ms/step - loss: 0.0837 - accuracy: 0.9679 - val_loss: 0.1710 - val_accuracy: 0.9489 - lr: 0.0010
Epoch 7/10
468/468 [==============================] - ETA: 0s - loss: 0.0645 - accuracy: 0.9766WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 175s 374ms/step - loss: 0.0645 - accuracy: 0.9766 - val_loss: 0.1708 - val_accuracy: 0.9465 - lr: 1.0000e-04
Epoch 8/10
468/468 [==============================] - ETA: 0s - loss: 0.0610 - accuracy: 0.9782WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 175s 374ms/step - loss: 0.0610 - accuracy: 0.9782 - val_loss: 0.1729 - val_accuracy: 0.9507 - lr: 1.0000e-04
Epoch 9/10
468/468 [==============================] - ETA: 0s - loss: 0.0572 - accuracy: 0.9794WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 175s 374ms/step - loss: 0.0572 - accuracy: 0.9794 - val_loss: 0.1685 - val_accuracy: 0.9489 - lr: 1.0000e-04
Epoch 10/10
468/468 [==============================] - ETA: 0s - loss: 0.0575 - accuracy: 0.9796WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Available
468/468 [==============================] - 175s 373ms/step - loss: 0.0575 - accuracy: 0.9796 - val_loss: 0.1712 - val_accuracy: 0.9489 - lr: 1.0000e-04
```

- ACCURACY

```
[40] score = model.evaluate(x_test, yval, batch_size=32)
     print()
     print("ACCURACY:",score[1])
     print("LOSS:",score[0])
```

```
130/130 [==============================] - 7s 56ms/step - loss: 0.2039 - accuracy: 0.9423

ACCURACY: 0.942307710647583
LOSS: 0.20393146574497223
```

8. CONCLUSION:

To conclude, our project meets its objective of counter the threat to deomocracty through the fake news and restore the public trust and give them access to real news for proper awareness and decision making. Our model uses the tensorflow and keras libraries for data preprocessing, text, embedding and classification. The data preprocessing has been done by tokenizing, label encoding, stemming, pad sequencing methods and word embedding is done by word2vec model from Gensim and text classification between true and fake by LSTM - long short term memory libabry. Our model accounts for an accuracy of 95%.

9. REFERENCES:

[1] https://www.ijraset.com/research-paper/paper-on-fake-news-detection-using-machine-learning

[2] https://ieeexplore.ieee.org/document/8546944

[3] https://www.sciencedirect.com/science/article/pii/S2667096820300070

[4] https://www.ijraset.com/research-paper/paper-on-fake-news-detection-using-machine-learning

[5] https://www.academia.edu/es/51346745/Fake_News_Detection_using_LSTM

[6] https://www.analyticsvidhya.com/blog/2021/07/detecting-fake-news-with-natural-language-processing

[7] https://towardsdatascience.com/predicting-fake-news-using-nlp-and-machine-learning-scikit-learn-glove-keras-lstm-7bbd557c3443?gi=4de371a5a91e

[8] https://www.sciencedirect.com/science/article/pii/S187705092101797X

[9] https://link.springer.com/article/10.1007/s42979-021-00775-6

[10] https://medium.com/analytics-vidhya/fake-news-detection-using-nlp-techniques-c2dc4be05f99