Fake News Detection Using Deep Learning

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology
in

Computer Science and Engineering/Information Technology

By

Aditi Garg (191379)

Under the supervision of

Dr. Pardeep Kumar

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

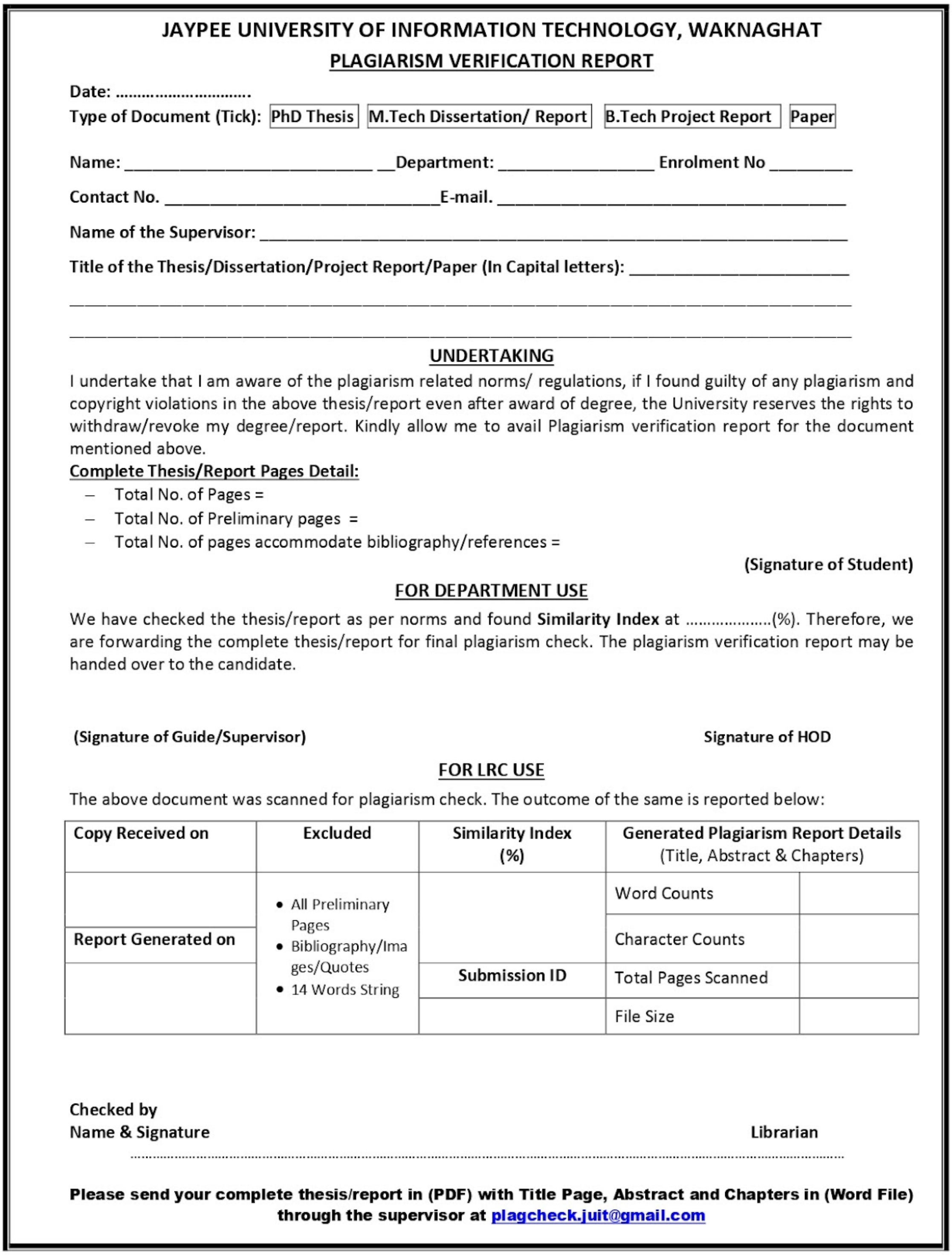
I hereby declare that the work presented in this report entitled **" Fake News Detection using Deep Learning"** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2023 to May 2023 under the supervision of **Dr. Pardeep Kumar (Associate Professor)**. I also authenticate that I have carried out the above-mentioned project work under the proficiency stream **Data Science.**
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Aditi Garg
(191379)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Pardeep Kumar
(Associate Professor)
Computer Science & Engineering and Information Technology
Dated:

PLAGIARISM CERTIFICATE



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: …………………………..

Type of Document (Tick): **PhD Thesis** **M.Tech Dissertation/ Report** **B.Tech Project Report** **Paper**

Name: ___________________________ __Department: _________________ Enrolment No _________

Contact No. ______________________________E-mail. ______________________________________

Name of the Supervisor: ____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): ________________________

____

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ………………..(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

……

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes it possible to complete the project work successfully.

I am really grateful and wish my profound indebtedness to Supervisor **Dr. Pardeep Kumar (Associate Professor)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of "**Data Science**" to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to **Dr. Pardeep Kumar (Associate Professor),** Department of CSE, for his kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Project Group No.: 84
Aditi Garg
(191379)

TABLE OF CONTENTS

List of Abbreviation

1. ANN: artificial neural network
2. LSTM: long-short term memory
3. CNN: convolutional neural networks
4. RNN: recurrent neural networks
5. NLP: natural language processing
6. AI: artificial intelligence
7. NLTK: natural language toolkit
8. APIs: Application programming Interfaces
9. RAM: random access memory
10. GPU: graphics processing unit
11. SGD: stochastic gradient descent
12. RMSProp: Root Mean Square Propagation
13. Max: maximum
14. Min: minimum

List of Figures

List of Tables

List of Graphs

Abstract

It is difficult to recognise reliable news sources due to the proliferation of false information as it’s easy to spread because of new sources including social media channels, news web journals, and online periodicals, Since more individuals now have access to the Internet thanks to the development of information and communication technology, this has altered how information is consumed therefore The need for computational tools that can provide insights regarding the fake quality of online content is growing.

In our everyday routine, we absorb news through a variety of sources, but occasionally it can be challenging to distinguish between fake and real news.

In this project, we will concentrate on text-based news and attempt to develop a model that will enable us to determine whether a given piece of news is legitimate or fraudulent. The primary goal is to identify bogus news, which is an easy concept for a standard text classification problem. A model that can distinguish between "real" and "fake" news must be developed.

The viability of employing deep learning algorithms to distinguish between real news and fake news on the Internet using only their text is investigated in this work. Two distinct neural network architectures are put forth in order to do that, one of which is based on CNN and other is a hybrid neural network architecture that combines the strengths of both CNN and LSTM.

Keywords—Fake news, CNN, LSTM, Hybrid approach, Neural network.

CHAPTER 1: INTRODUCTION

1.1 Introduction

Online data is generated in enormous quantities every day in the age of technology. Fake news, on the other hand, has invaded the Internet in an unprecedented amount, intended to bring attention to itself and sway people's beliefs and actions. Spreading maliciously false information that leads to reader confusion, it is now extremely easy to obtain information and share it through social media platforms, hence making It's challenging and not simple. to distinguish new accounts purely on the content of fresh posts.

The media has consequently begun to respond to the new developments. Some, for instance, have started to give more importance to their online presence or have made the decision to begin utilizing new avenues of distribution, videos or podcasts.

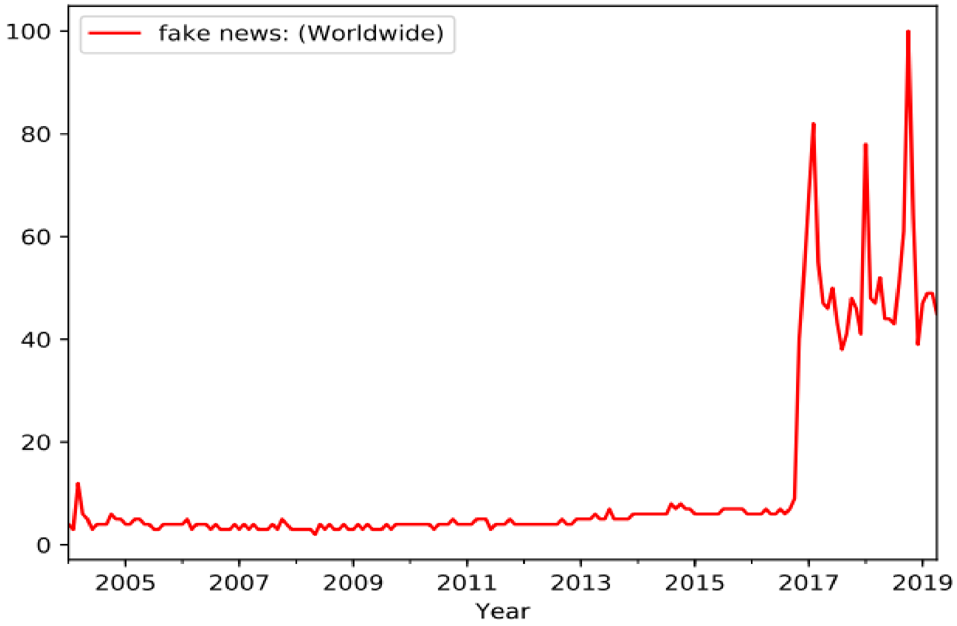


Figure 1.1: ‘Fake News’ in Google Trends

Finding a means to make these new different channels of distribution, that have always been available for the consumer, profitable is their current project. Large-scale fabrications of fact, Cascade have negative repercussions in the business, advertising, and stock-share sectors that progressively get harsher with duration.

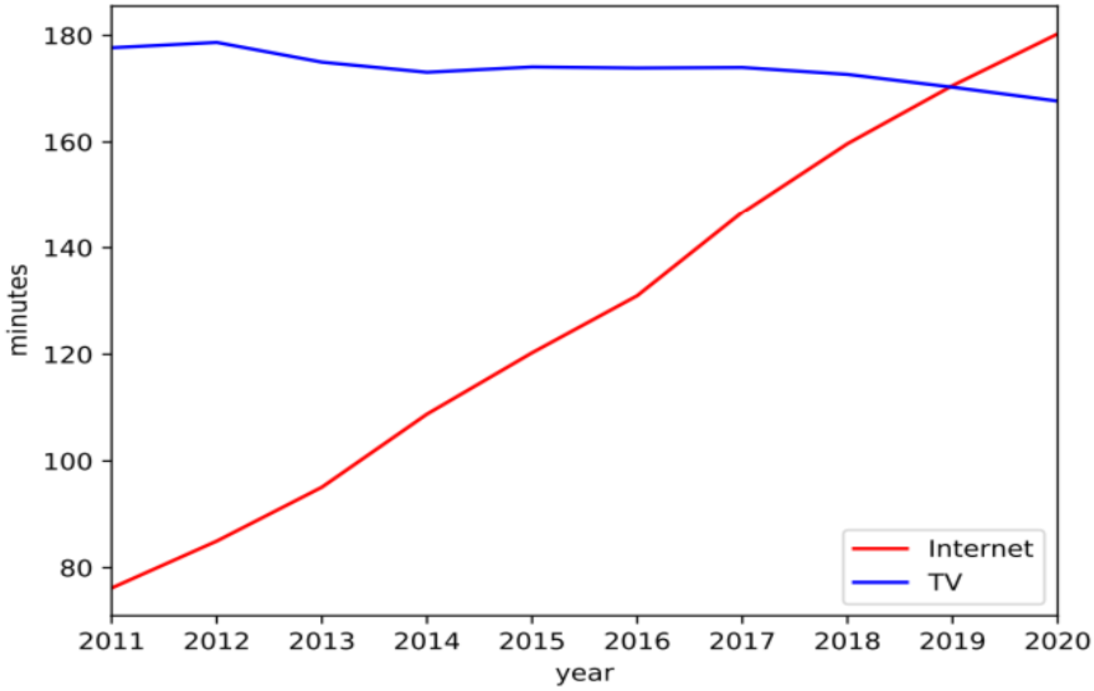


Figure 1.2: Average daily media consumption worldwide

The top internet companies, like Google, Facebook, etc. are aware of this risk and have already begun creating systems to spot fake news on their sites. However, even if the solutions are developing quickly, the issue is still quite complex and requires more research.

The major objective of the research is to establish a number of deep neural network-based models that can be used for identifying and classifying fake news so that people can evaluate the accuracy of the material they read and, to the greatest extent possible, steer clear of prejudice and misunderstandings.

Convolutional neural networks (CNN) and LSTM, two deep learning models, successfully in a variety of NLP tasks that are comparable to detecting false news, including evaluating the semantic relatedness of texts. A deep neural network transforms the text sequence into a fixed length vector representation, which is then used to calculate the significance of each headline-body pair in two textual sequences.

1.2 Problem Statement

In this day and age, it is extremely difficult to decide whether the news we come across is real or not. There is an acute need to create a system for detecting fake news using deep learning models. In a specific situation, the algorithm must be able to identify bogus news.

Formalizing the problem to be solved by presenting it as a news story with a certain set of attributes (ie. id, title, author, text, label, ...), It seeks a:

$$f(a)= \begin{cases} 0 & \text{if } a \text{ is fake} \\ 1 & \text{if } a \text{ is true} \end{cases}$$

1.3 Objective

In this project, we will concentrate on text-based news and attempt to create a model that will allow us to determine whether a given piece of news is legitimate or fraudulent.

The goal of employing deep learning models for fake news identification is to create an algorithm that can reliably and accurately distinguish between news articles that are fraudulent and those that are genuine.

In recent years, fake news has grown to be a serious issue, especially on social media platforms where it can spread quickly. Traditional techniques of fact-checking and verification can take a lot of time, and they might not be able to keep up with how quickly false information is being created and disseminated. Therefore, automated methods to identify fake news are required.

The detection of fake news has shown encouraging results when using deep learning models like CNNs (Convolutional Neural Networks), LSTMs (Long Short-Term Memory), and Hybrid CNN+LSTM. These algorithms analyze the text using sophisticated approaches in order to find patterns that can distinguish between phony and real news stories.

These models are being used with the intention of offering a quick, scalable, and accurate way to recognize and categorize fake news, which can aid in the fight against the spread of false information and shield the general public from its negative impacts. We can automate fake news identification and make it more dependable, effective, and available by employing deep learning models.

In general, the goal of utilizing deep learning models for fake news detection is to create an algorithm that can reliably identify and categorize bogus news in real-time, helping to stop the spread of disinformation and guaranteeing that the general public receives accurate and trustworthy information.

1.4 Motivation

Fake news is a very persistent problem in the modern digital era, when there are dozens of information sharing websites where incorrect information can propagate. construction of artificial intelligence (AI), which brings with it Fake news has become a bigger issue as a result of artificial bots that might be used to create and spread it. The problem is serious because many people believe what they read online, and those who lack expertise or are unfamiliar with digital technologies run the risk of being easily duped. Fraud that may result from spam or harmful emails and messages is another issue. Therefore, it is a persuasive reason to accept the issue and embrace the task of reducing crime, political turmoil, grief, and opposing the propagation of false information.

Addressing the growing issue of fake news in the current digital era is the driving force behind this study paper. Misinformation and propaganda may significantly affect society, affecting everything from elections to inciting violence. Automated false news detection can lessen the negative consequences of fake news while halting its spread.

1.5 METHODOLOGY USED

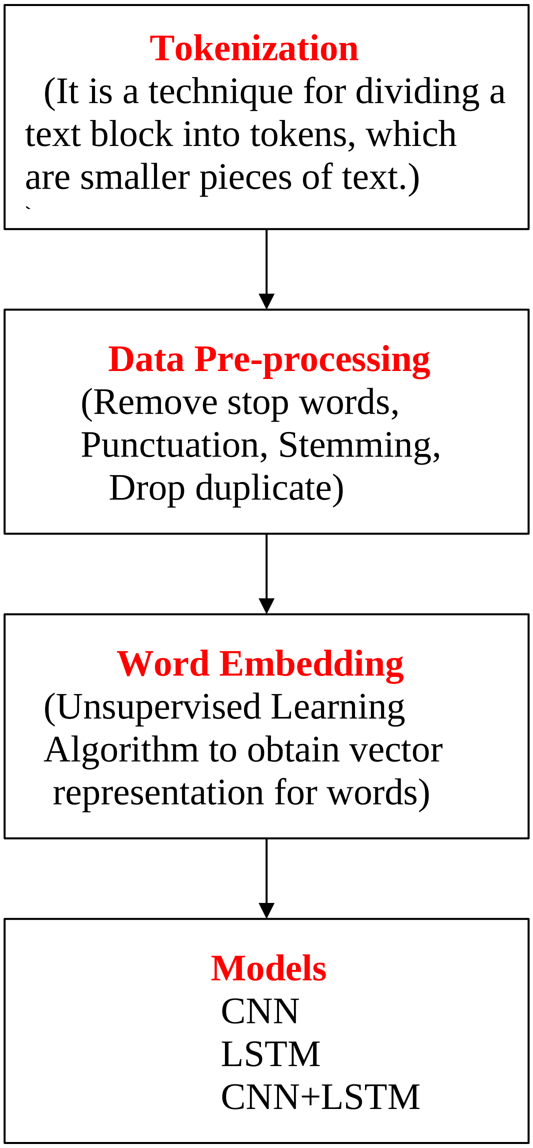

Tokenization
(It is a technique for dividing a text block into tokens, which are smaller pieces of text.)
Data Pre-processing
(Remove stop words, Punctuation, Stemming, Drop duplicate)
Word Embedding
(Unsupervised Learning Algorithm to obtain vector representation for words)
Models
CNN
LSTM
CNN+LSTM

Figure 1.3: Methodology used

1.6 Organization

The remaining part of the academic paper is ordered as follows: Chapter 2 we have presented the literature survey which depicts the various approaches used by authors to create an Image Captioning model.

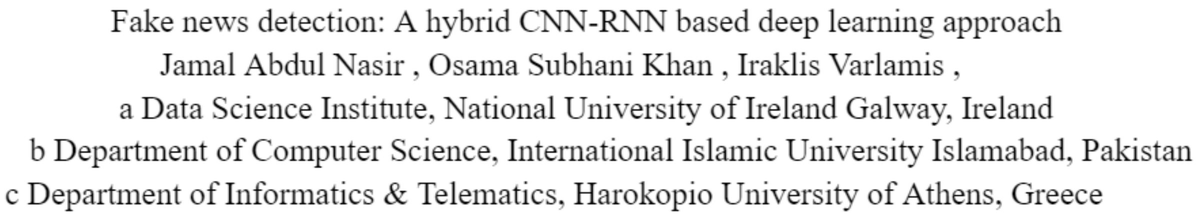
Chapter 3 highlights the methodology and system development of the project. It represents various computational, experimental and mathematical concepts of the project. Also, we have focused on the software and hardware platforms needed for implementing the model. Also, we have shown the required dataset and its related information.

In chapter 4 we have presented the performance analysis of the project which specifies the accuracy of the project. And also made a comparison between the model used in this project.

Chapter 5 presents the conclusions of the project and the observations seen in the results. It also provides the applications of the project and the future scope of the same.

CHAPTER 2: LITERATURE SURVEY

The first paper that we referred to is-



Fake news detection: A hybrid CNN-RNN based deep learning approach

Jamal Abdul Nasir , Osama Subhani Khan , Iraklis Varlamis ,

a Data Science Institute, National University of Ireland Galway, Ireland

b Department of Computer Science, International Islamic University Islamabad, Pakistan

c Department of Informatics & Telematics, Harokopio University of Athens, Greece

Abstract

The advent of social media has made it possible for people to disseminate knowledge more easily, cheaply, and using the least number of filters ever employed. This made the long-standing problem of fake news worse, which is now a serious worry because of the harm it causes to communities. To stop the emergence and spread Automatic detection methods of bogus news have been studied. These methods rely on machine learning and artificial intelligence.

Deep learning techniques have lately made substantial they have made progress in solving challenging natural language processing problems, giving them a good option for identifying bogus news as well. In order to classify fake news, this study introduces a novel hybrid deep learning model that blends convolutional and recurrent neural networks. On fake news datasets **(ISO and FA-KES),** the model was successfully validated. obtaining detecting outcomes that outperform other non-hybrid baseline techniques by a wide margin. The results of further investigation on the applicability of the suggested model to various datasets were encouraging.

Conclusion

There is still a tonne of room for experimentation when it comes to fake news detection, models could become more accurate and efficient as a result of the finding of new knowledge of the characteristics of fake news. To the best of our knowledge, this research is also the first to propose generalizing fake news detection methods. These models, as demonstrated in this paper, typically perform well on a particular dataset but do not generalize well. By taking into account the generalization of false news detection models, new boundaries can be investigated for our analysis to come. Combining traditional models with task-specific feature engineering strategies might also be advantageous.

In general, Artificial neural networks appear to hold potential for use in identifying bogus news. More complicated neural network architectures will be taken into consideration in addition to CNN and RNN.

The efficiency of the model is improved by the authors' proposal in this research to combine (CNN) and (RNN).

Result

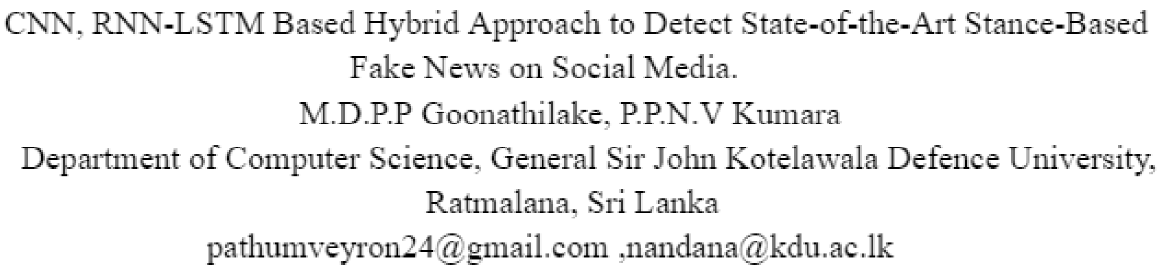
Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
50	48	48	46

Table 2.1: Result of research paper_1

Disadvantage

Poor generalization is revealed by cross dataset validation results, 0.50 Despite employing a massive training corpus, it performs poorly on a separate fake news dataset with the same structure.

The next paper referred to was-



CNN, RNN-LSTM Based Hybrid Approach to Detect State-of-the-Art Stance-Based Fake News on Social Media.

M.D.P.P Goonathilake, P.P.N.V Kumara

Department of Computer Science, General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka

pathumveyron24@gmail.com ,nandana@kdu.ac.lk

Abstract

The transmission of incorrect False information is spread through online social media or conventional news sources., a recent phenomenon. Today, fake news is easily produced, disseminated over a wide range of social media channels, and has a significant effect on reality. It is essential to create effective the reasons why fraudulent information on social media platforms is successful in misleading viewers, as well as algorithms and methods for early identification of it. Today's most widely utilized methods of research for spotting false Information is supported by machine learning, deep learning, feature extraction, graph mining, and image and video analysis., as well as recently developed data and online services. Finding an appropriate technology that can quickly identify erroneous information is therefore urgently needed. It was suggested that to identify incorrect information from this study, Use of the CNN and RNN-LSTM models. Using the NLTK toolkit, stop words, grammar, and special characters were first removed from the text. The text is then tokenized and preprocessed using the same toolkit. Since then, the preprocessed text has been improved with GloVe word embeddings. Convolutional and max-pooling layers used in the CNN model to extract higher-level text input features. Word sequences' long-term dependencies are extracted from the RNN LSTM model. The suggested method makes use of dense layer dropout method to boost the hybrid model's effectiveness even more. The findings of the recommended hybrid model, Modern traditional models using the Binary Cross-Entropy loss function and Adam optimizer demonstrate that the suggested CNN, RNN-LSTM based hybrid method achieves the greatest accuracy of 92%, outperforming most other hybrid models.

Conclusion

The hybrid neural network architecture proposed after 10 iterations using the Adam optimizer and the Binary Cross-Entropy loss function obtained about 92% results in comparison to data and studies on stance-based false news identification.

We may therefore draw the conclusion that the proposed hybrid technique using CNN, RNN, and LSTM, which outperforms the majority of current classical models, obtains the greatest accuracy of 92%.

Regarding the proposed hybrid model's future work, we anticipate running it on transformers similar to Bert and then conducting some tests with federated learning APIs. In light of this, we intend to post the dataset in Kaggle datasets for additional research. Last but not least, the outcomes of this study will be updated on a public GitHub repository and receive updates on new experiments performed on the dataset periodically.

In this paper, the authors have proposed in order to increase accuracy, a hybrid method was used that concentrated on NLP techniques including Word embeddings, tokenization, preprocessing, and a variety of deep learning models, including CNN, LSTM.

Result

Comparison Of Result	CNN Model	RNN Model	Hybrid Model
Accuracy	0.90	0.90	0.92
Loss	0.22	0.15	0.18

Table 2.2: Results of research paper_2

Disadvantage

After 10 epochs It achieved approximately 92% accuracy from the proposed hybrid neural network design.

The Third paper is

Fake News Detection Using A Deep Neural Network
Rohit Kumar Kaliyar Computer Science
& Engineering Bennett University Greater Noida, India
rk5370@bennett.edu.in

Abstract

Using social networks to get news is like wielding a two-edged sword. On the one hand, it is user-friendly, quick to access, easy to use, and suitable for social sharing.

Every minute, new information is added with the potential for several perspectives on a single story. On the other hand, several social media sites govern news depending on individual preferences or interests. The term "fake news" is used to describe inaccurate information or news that has been modified and distributed on social media with the intention of hurting a person, business, or organization. The spread of misleading information required the creation of computer algorithms to recognize it. The goal of identifying fake news is to help users expose various forms of false information. We can decide if the news is accurate or false based on examples of true or false reporting from the past. There are several ways that we might use social media to access misleading news. Two contributions are provided. To categorize the datasets, we employ NLP, machine learning, and deep learning models. By incorporating false news categorization and current machine learning models, we produce a thorough audit of detecting fake news.

Conclusion

Analysis of socially pertinent data is the process of detection of false news. to ascertain if the situation is real. Numerous Nave and other machine learning algorithms were evaluated for this project.

Shallow Convolutional Neural Networks from Deep Learning, Bayes, K Nearest Neighbors, and Random Forest are examples of decision-making algorithms. These types of neural networks include convolutional neural networks with short-term memory (CNN-LSTM), very deep convolutional neural networks (VDCNN), gated recurrent unit networks (GRU), and convolutional neural networks with long-term memory (CNN-LSTM).

We also looked at feature extraction and features' advantages. In our study, TF-IDF characteristics were retrieved and used similarly to n-gram model. We also looked at word embeddings effectiveness, and deep neural network word2vec.

Result

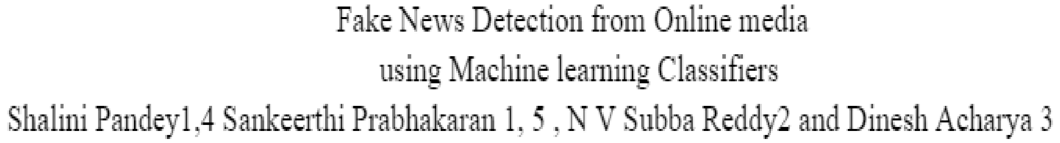
Models	CNN+LSTM	Naive Bayes	Decision Tree	Random Forest	K-Nearest Neighbor
Precision	0.97	0.90	0.75	0.72	0.55
Recall	0.96	0.90	0.74	0.71	0.54
F1-Score	0.97	0.90	0.73	0.71	0.50

Table 2.3: Results of research paper_3

Disadvantage

In this study, the authors suggest implementing our models using NLP, machine learning, and deep learning methods while comparing which models will provide the highest accuracy. LSTM and CNN were coupled in the execution. Accuracy dropped somewhat to 97.3%. From 98.3%.

The fourth paper we referred to is



Fake News Detection from Online media
using Machine learning Classifiers
Shalini Pandey1,4 Sankeerthi Prabhakaran 1, 5 , N V Subba Reddy2 and Dinesh Acharya 3

Abstract

Technology development has caused a shift in how people consume news, ranging from print to social networks. Significant contributors to this change in news consumption are accessibility and convenience. The spread of "Fake news" online, where there is less oversight, has presented a new difficulty as a result of this transformation. This problem has been handled in this study using the idea for machine learning with the aid of data pre-processing, algorithms like Naive Bayes, Decision Trees, Support Vector Machine (svm), K-Nearest Neighbor, and Logistic Regression Classifiers have improved their efficiency in distinguishing between fake and true news in a given dataset. Along with the results, a comparison of how these classifiers operate is also provided.

Conclusion

Given that the data required is text and the article includes a variety of features that should be taken into account, discussing the target, such as the categorization of news, is a challenging issue even when utilizing classifier approaches.

It has been discovered that utilizing the Word2vece process of computing with text takes time. In addition, classifiers with strong accuracy reports are simpler to use. Word2Vector is typically not advised due to its high RAM and disc usage, although it does provide vectorizing data using a logical relation. The project might be improved upon to become a practical programme that could evaluate the veracity of any input, regardless of language.

In this paper, the authors have proposed Efficient is the NLTK toolkit, which includes a number of NLP-focused tools and a collection of libraries. even the machine learning methods for classifying, regressing, and grouping data.

Result-

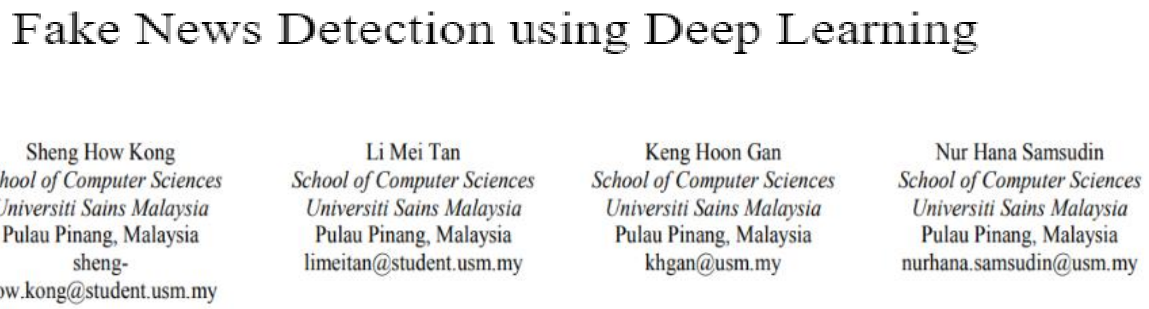
Model	KNN	Logistic Regression	Naive Bayes	Decision Tree	SVM classifiers
Accuracy	89.98%	90.46%	86.89%	73.33%	89.33%

Table 2.4: Results of research paper_4

Disadvantage

It is discovered through the use of Word2vec that processing text for computing takes time. Word2Vector is typically not advised due to its high RAM and disc usage, although it does provide logical relation for turning data into vectors.

The fifth paper we referred to is.



Fake News Detection using Deep Learning

Sheng How Kong
School of Computer Sciences
Universiti Sains Malaysia
Pulau Pinang, Malaysia
sheng-how.kong@student.usm.my

Li Mei Tan
School of Computer Sciences
Universiti Sains Malaysia
Pulau Pinang, Malaysia
limeitan@student.usm.my

Keng Hoon Gan
School of Computer Sciences
Universiti Sains Malaysia
Pulau Pinang, Malaysia
khgan@usm.my

Nur Hana Samsudin
School of Computer Sciences
Universiti Sains Malaysia
Pulau Pinang, Malaysia
nurhana.samsudin@usm.my

Abstract

False information has reached a wider audience than ever before in the digital age. The growth of social media and direct messaging platforms is the primary reason. There are many different, creative, and creative ways to identify fake news reports. This project aims to develop deep learning algorithms for text analytics and using natural language processing (NLP) for detecting false information based on article titles or news outlets. The goal of the study's approach is to lessen the bad user experience associated with receiving inaccurate information from unreliable sources on genuine social media sites.

NLP techniques preprocess text using Prior to vectorizing text into N-grams or sequential vectors using terms frequencies inverse document occurrence (TF-IDF) or one-hot encoding, regular expression, tokenization, lemmatization, and stop words deletion are used. TensorFlow is chosen as the framework to be utilized and developed in Keras deep learning libraries because it has a sizable community and a lot of commits on the TensorFlow GitHub repository, both of which can be sufficient to build deep learning neural network models. The models' results demonstrate that, although requiring less computation time to perform well, models trained with news content can nevertheless beat models trained with other types of content. Models fed with N-gram vectors are also outperform models fed with sequence somewhat overall.

Conclusion

Finally, the most effective neural network model developed for this study can reliably identify bogus news with up to 90.3% accuracy and 97.5% recall, with very few errors. Because N-gram vectors using TF-IDF will depend on term frequency as well as a weight score that prioritizes more important terms, models trained with N-gram vectors outperform models trained with sequence vectors slightly. Due to their quick computation time and high recall rate (low mistake rate), models trained using news titles are highly suited for usage in social media apps where users are expected to respond quickly to any updates or incoming messages. Any communication that spreads false information can be stopped with a quick calculation. To accurately identify false news and prevent people from sharing it, models with higher accuracy and recall that have been trained on news content would be preferable. On the other hand, social media applications with infrequently updated feeds do not always need rapid processing times. The Keras neural network models can finally be enhanced to attain even higher recall and accuracy by adjusting the settings. The effectiveness of false news recognition with NLP (LSTM) can also be improved by using recurrent neural networks (RNN) and a long short-term memory approach. More analysis of news images, videos, and articles can be done in the future to enhance the models. Similar methods or strategies might be employed to create the models using news data obtained in Malaysia in order to further deploy this solution in Malaysia. It is wise to carry out more research and experimenting on Malay and Chinese news.

In this paper, the authors have proposed to develop deep learning algorithms for text processing and use natural language processing (NLP) approaches to identify false news depending on the headline or the sources.

Result

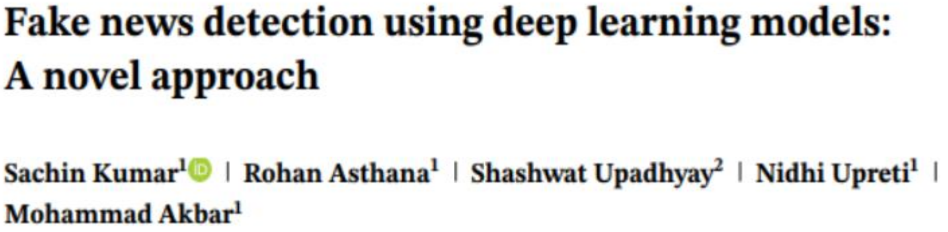
Model	Accuracy (%)	Recall (%)
1	77.3	89.7
2	90.3	97.5
3	74.8	89.8
4	90.0	94.0

Table 2.5: Results of research paper_5

Disadvantage

versus models conditioned on news content, those with news headline experience need to calculate things much faster. lengthier calculation times, news-based model training.

The last paper we referred to is -



Fake news detection using deep learning models: A novel approach

Sachin Kumar[1] | Rohan Asthana[1] | Shashwat Upadhyay[2] | Nidhi Upreti[1] | Mohammad Akbar[1]

Abstract

Controlling the propagation of false data and reducing reliance on data gained from such sources are now essential due to the ever-increasing use of social media. Because readers' interactions with fake and misleading news help spread it on a personal basis, social media platforms are under constant pressure to come up with solutions that are successful. The False information being circulated has an adverse effect on how people see a significant activity, thus it must be addressed in a contemporary manner. By accumulating 1356 information instances from various people via Tweets and news websites like PolitiFact, we create multiple data for the real and fake news items in this study. Our study compares a number of cutting-edge technologies, including CNN models (CNNs), long short-term memory (LSTMs), ensemble methods, and attentive processes. The CNN plus an ensembled bidirectional LSTM network with attentive mechanism, according to our findings, had the maximum precision of 88.78%, while Koetal. attempted to identify bogus news and had a detection rate of 85%.

Conclusion

With the help of this research, we can counteract incorrect information on a big scale and mitigate its severe impacts. A thorough examination of this topic can be conducted using the framework and helpful tools provided by our study. Moving forward, digital media education initiatives can aid in educating the public on urgent topics in social media and reducing misunderstandings about them.

Finally, we want to discuss the limitations of this study. First, due to limited resources, this research's primary focus was on the sentiments of news reports rather than continuously examining the veracity of the news sources themselves. Keeping track of the author of incorrect reports and including them as a research parameter might further improve accuracy. Due to the comparative nature of our investigation, during its transition from actual news to fake news, our classification techniques weren't able to recognize the linguistics shift transmission. Moreover, by utilizing additional freshly created state-of-the-art architectures, this research may be broadened to reach greater accuracy.

In this paper, the authors have proposed to Utilize tools like neural network models, long short-term memories, and convolutional neural networks (CNNs).

Disadvantage

Due to a lack of resources, this research concentrated mostly on the opinions expressed in news articles rather than continuously examining the reliability of the news sources themselves.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 Computational

High configuration GPUs are used for training the model. These are available online and also available on one’s system. The training time is dependent on the GPU. GPUs with higher memory like 4-16 GB are recommended for such applications. Software’s like Jupyter notebook are preferred but applications like PyCharm and VScode can also be used along with python libraries like NumPy, Keras and TensorFlow.

For our project, we used Jupyter to write and execute code in python language, and it is well suited to machine learning, deep learning, data analysis and education.

Experiments were run on Jupyter with Intel® Xeon ® processor at 2.20GHz using 12.72 GB of RAM coupled with a Nvidia Tesla T4.

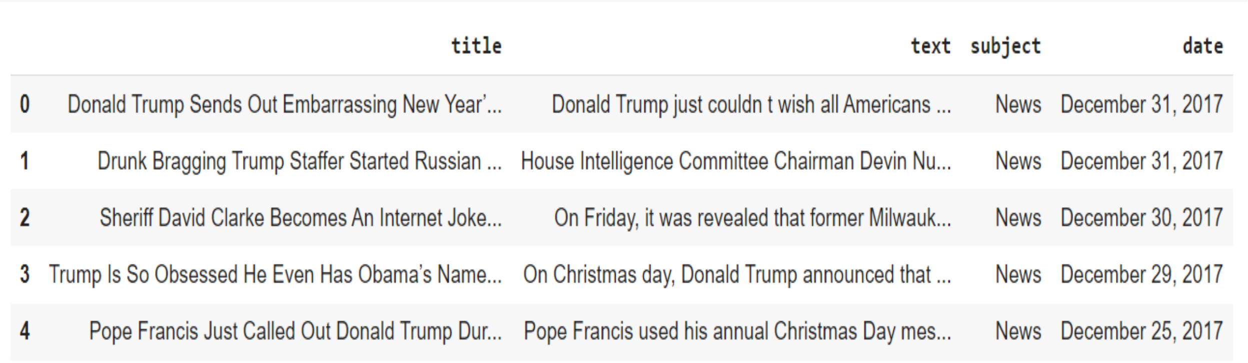
3.2 Dataset Collection

Dataset is collected from Kaggle that is-**Fake and real news dataset**

Two files are used-:

Fake.csv-consist of 4 columns

- Title-The title of the article
- Text-The text of the article
- Subject-The subject of the article
- Date-The date at which the article was posted

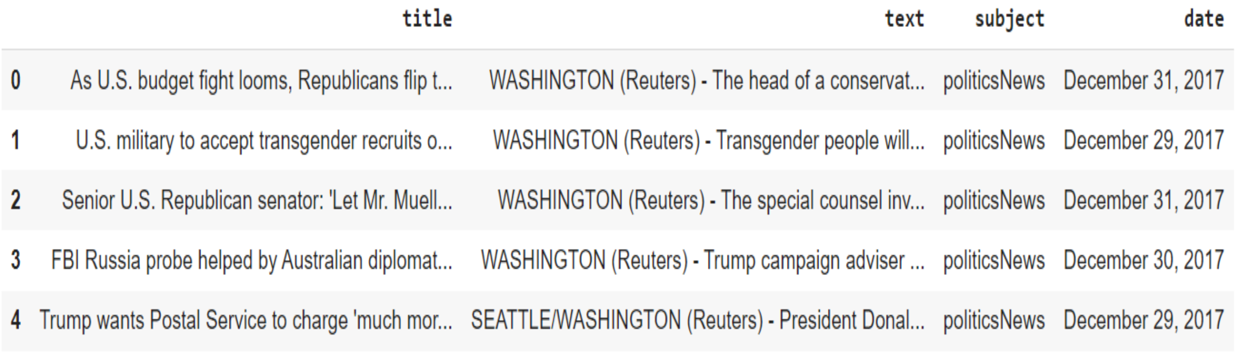


	title	text	subject	date
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
2	Sheriff David Clarke Becomes 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

Figure 3.1: SCREENSHOT of Fake dataset

True.csv-consist of 4 columns

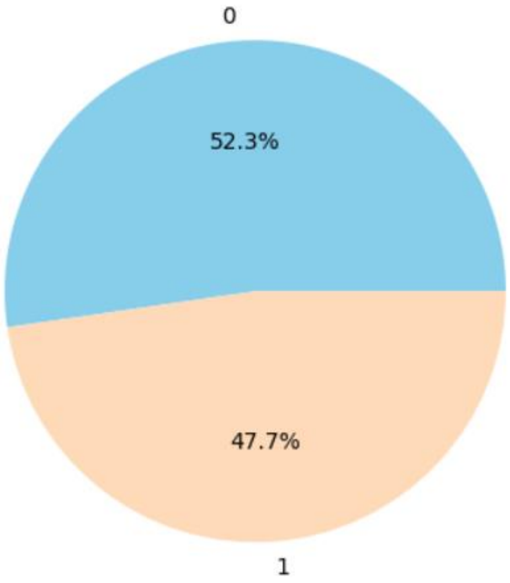
- Title-The title of the article
- Text-The text of the article
- Subject-The subject of the article
- Date-The date at which the article was posted



	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

Figure 3.2: SCREENSHOT of True dataset

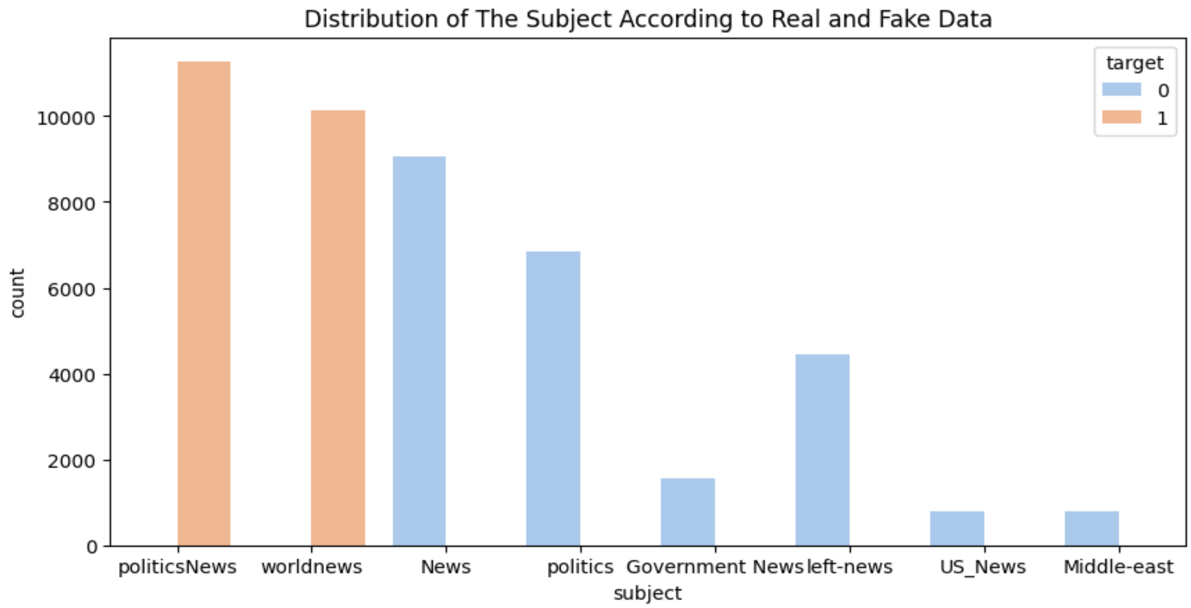
The number of fake news (0) are-23481 and number of True news (1) are-21417



Graph 3.1: Pie Chart for Fake and True News

- It consists of variety of news-

politics News	11272
world news	10145
News	9050
politics	6841
left-news	4459
Government News	1570
US_News	783
Middle-east	778

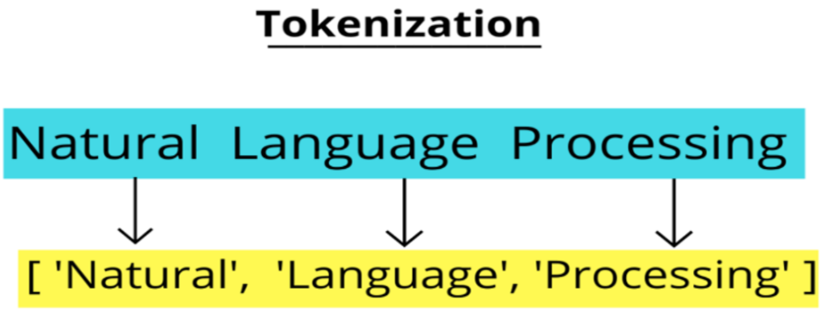


Graph 3.2: Bar Graph for distribution of the Subject According to Real and Fake data

3.3 Tokenization

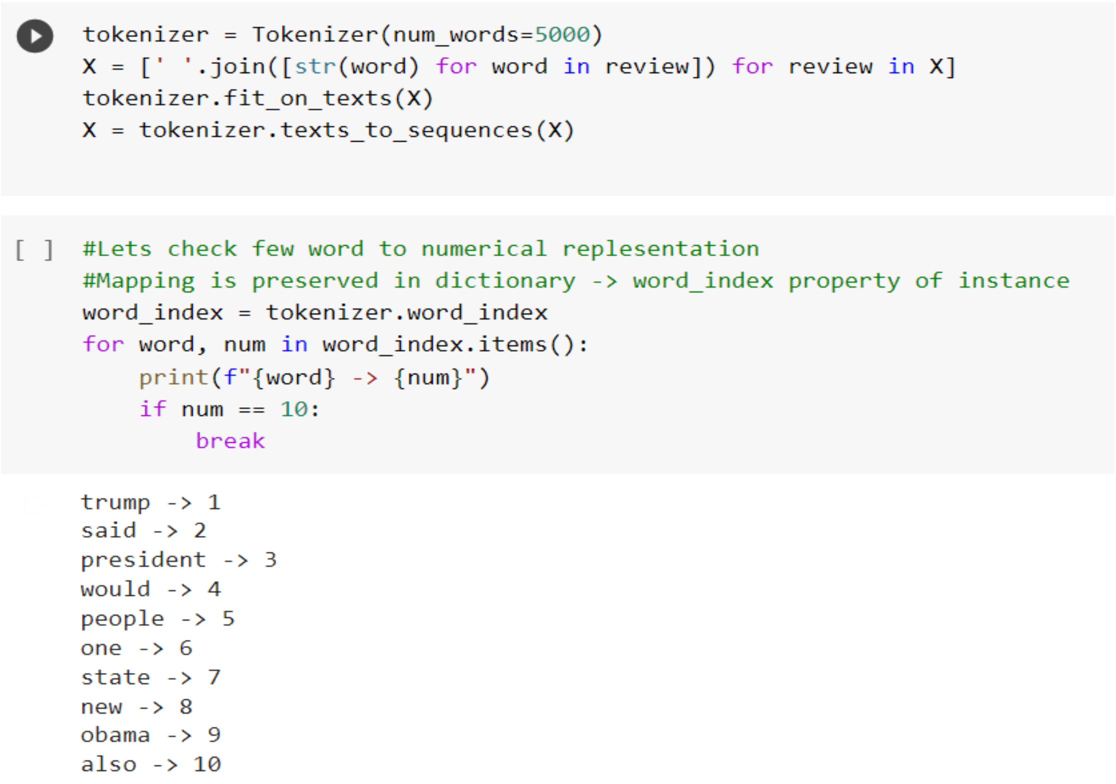
Tokenization is the process of separating natural language text and unstructured data into units of data that may be viewed as discrete chunks. Token occurrences in a document can be used directly as a vector to express the text.

Figure 3.3: Tokenization


Tokenization
Natural Language Processing
['Natural', 'Language', 'Processing']

Why do we need Tokenization?

The initial stage in every NLP pipeline is tokenization. It significantly affects the remainder of your pipeline. We could wish to count the number of times each word appears in the provided text by dividing it up into tokens.



```
tokenizer = Tokenizer(num_words=5000)
X = [' '.join([str(word) for word in review]) for review in X]
tokenizer.fit_on_texts(X)
X = tokenizer.texts_to_sequences(X)
```

```
[ ]  #Lets check few word to numerical replesentation
     #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
```

Figure 3.4: Tokenizing is done

3.4 Data Preprocessing

To apply machine learning or deep learning algorithms to text data, extra preprocessing is needed. Text data can be transformed using a number of methods that are frequently used to create models-ready formats. The headlines and news items are subjected to the data preparation methods that we describe below.

1.Stop Word Removal

2.Punctuation Removal

3.Lemmatization

Stop words

Stop words are those that are usually overlooked while analyzing natural language. Even though these are some of the most widely used words in any language, the text does not gain much information from them. The words “a", "the", "is”, “very" and "so" are some examples of stop words in English.

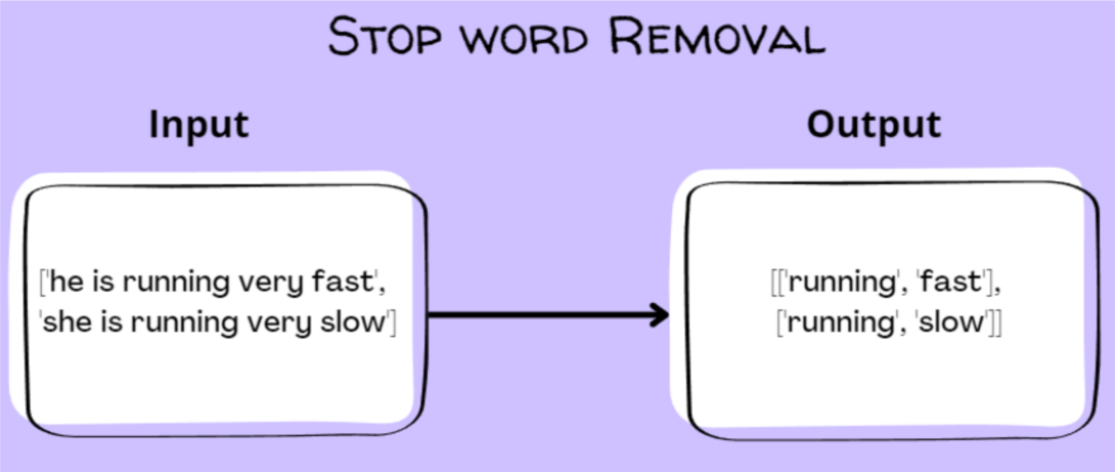
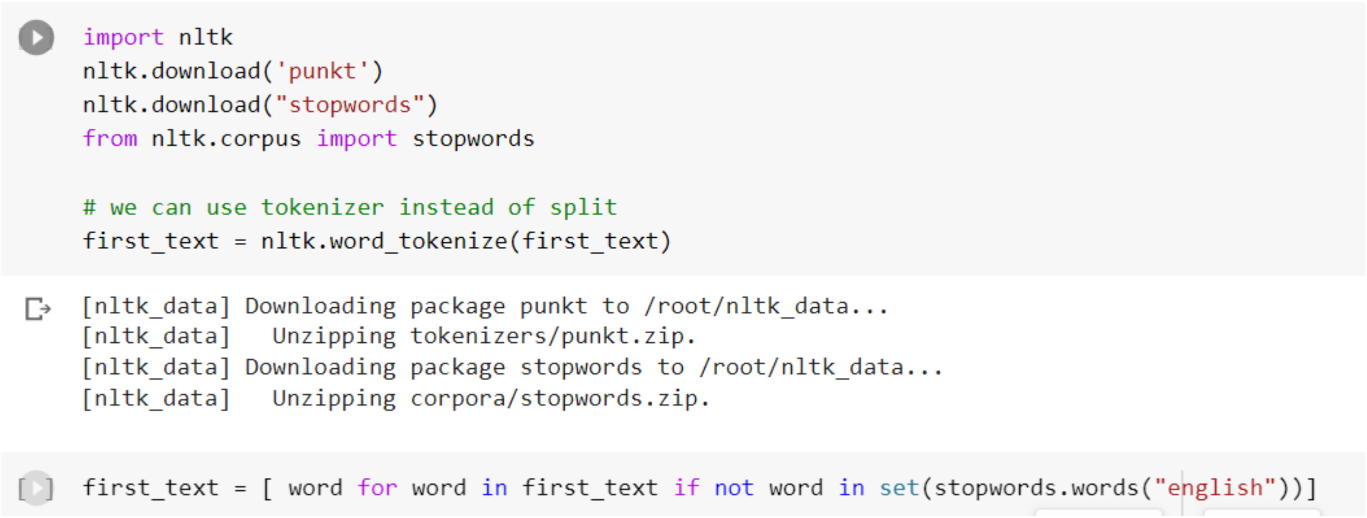

STOP WORD REMOVAL
Input
Output
['he is running very fast', 'she is running very slow']
[['running', 'fast'], ['running', 'slow']]

Figure 3.5: Stop word removal

Why do we remove stop words?

There are many stop words in every human language. We can make our text more centered on the valuable information by getting rid of these words and cutting out the low-level information. To put it another way, we claim that the model we create for our research does not demonstrate any negative effects as a result of the removal of such sentences.

The dataset's size and, subsequently, the training time are minimized by excluding stop words.

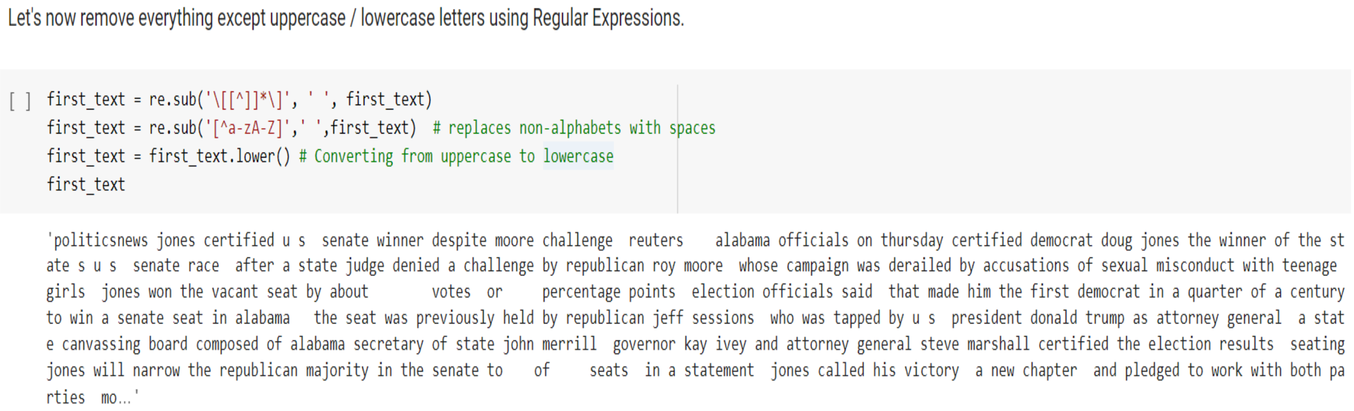


```
import nltk
nltk.download('punkt')
nltk.download("stopwords")
from nltk.corpus import stopwords

# we can use tokenizer instead of split
first_text = nltk.word_tokenize(first_text)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
[ ] first_text = [ word for word in first_text if not word in set(stopwords.words("english"))]
```

Figure 3.6: Stop word removal code



Let's now remove everything except uppercase / lowercase letters using Regular Expressions.

```
[ ] first_text = re.sub('\[[^]]*\]', ' ', first_text)
    first_text = re.sub('[^a-zA-Z]',' ',first_text)  # replaces non-alphabets with spaces
    first_text = first_text.lower() # Converting from uppercase to lowercase
    first_text
```

```
'politicsnews jones certified u s  senate winner despite moore challenge  reuters    alabama officials on thursday certified democrat doug jones the winner of the st
ate s u s  senate race  after a state judge denied a challenge by republican roy moore  whose campaign was derailed by accusations of sexual misconduct with teenage
girls  jones won the vacant seat by about        votes  or     percentage points  election officials said  that made him the first democrat in a quarter of a century
to win a senate seat in alabama   the seat was previously held by republican jeff sessions  who was tapped by u s  president donald trump as attorney general  a stat
e canvassing board composed of alabama secretary of state john merrill  governor kay ivey and attorney general steve marshall certified the election results  seating
jones will narrow the republican majority in the senate to    of     seats  in a statement  jones called his victory  a new chapter  and pledged to work with both pa
rties  mo...'
```

Figure 3.7: Removal of Punctuation Marks and Special Characters

Lemmatization

Lemmatization is a method of natural language processing that includes stripping a word down to its most basic or dictionary form, or lemma. It is a crucial component of text preprocessing and is frequently applied to tasks like text categorization, sentiment analysis, and information retrieval.

Depending on the tense, number, and context of a word, it can have various forms in English. For instance, the word "child" might be written as "children" or "child's", while the verb "run" can be written as "running", "ran", or "runs". All these distinct spellings of the same word can be reduced to their respective base forms—"run" and "child"—by using lemmatization.

Lemmatization is distinct from stemming, another text preparation method that entails stripping words of their suffixes to return them to their original form. Although stemming is quicker and easier than lemmatization, it may produce non-words or incorrectly spelt words, which may impair the accuracy of subsequent tasks.

A lemmatizer, a tool or library that maps words to their base form based on their part of speech, is used in lemmatization. The word "running," for instance, may be lemmatized to "run" if it is understood to be a verb, but it could also be lemmatized to "running" if it is understood to be a noun.

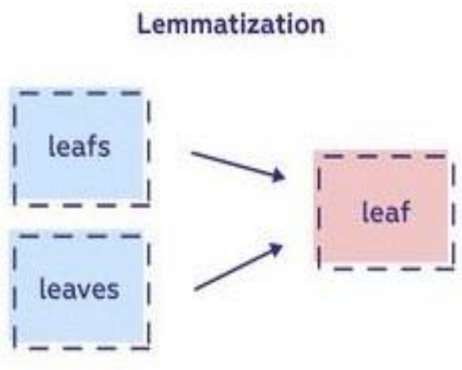
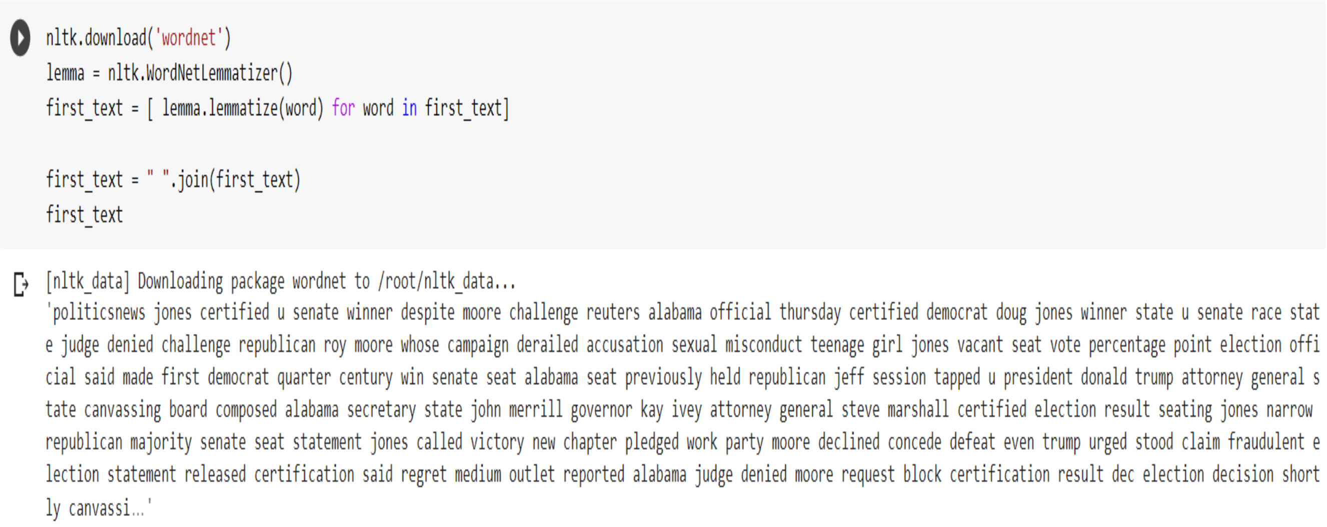


Figure 3.8: Lemmatization



```
nltk.download('wordnet')
lemma = nltk.WordNetLemmatizer()
first_text = [ lemma.lemmatize(word) for word in first_text]

first_text = " ".join(first_text)
first_text
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
'politicsnews jones certified u senate winner despite moore challenge reuters alabama official thursday certified democrat doug jones winner state u senate race stat
e judge denied challenge republican roy moore whose campaign derailed accusation sexual misconduct teenage girl jones vacant seat vote percentage point election offi
cial said made first democrat quarter century win senate seat alabama seat previously held republican jeff session tapped u president donald trump attorney general s
tate canvassing board composed alabama secretary state john merrill governor kay ivey attorney general steve marshall certified election result seating jones narrow
republican majority senate seat statement jones called victory new chapter pledged work party moore declined concede defeat even trump urged stood claim fraudulent e
lection statement released certification said regret medium outlet reported alabama judge denied moore request block certification result dec election decision short
ly canvassi...'
```

Figure 3.9: Lemmatization code

3.5 WORD EMBEDDING

It is a technique for presenting printed words and materials. A word is represented by a lower-dimensional numeric vector input called a word vector or word embedding. It allows for comparable representations of words with comparable meanings. Additionally, they can loosely convey meaning. A word vector can represent 100 different qualities with its 100 values.

Features: Any element that connects words to one another. For instance, age, occupation, fitness, and sports. Values for each word vector relate to these features.

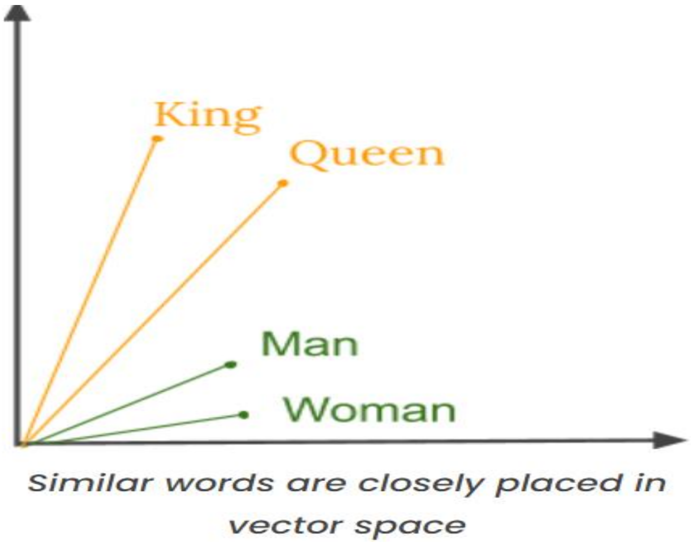


Figure 3.10: words in vector space

Goal of Word Embeddings

1.To significantly reduce the dimensionality.

2. Use a letter to foretell the words surrounding it.

3. The inter word relationship must be recorded.

How are Word Embeddings used?

They generate input for models of machine learning.
words —–>numeric representation —–> Used for training.
To represent or display any usage patterns that may have been in the corpus used to train them.

Implementations of Word Embeddings:

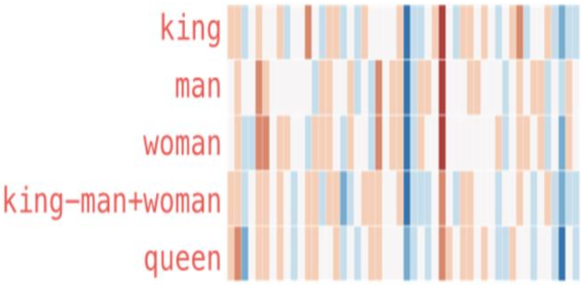
Word embeddings are a technique for taking the features from text and putting them into a machine learning model that can work with text data. They make an effort to maintain semantic and syntactic information. The word count in a sentence is used by techniques like Bag of Words (BOW), Count Vectorizer, and TFIDF, although no syntactic or semantic data is saved. The amount of vocabulary elements in these methods determines the size of the vector. If the majority of the elements are 0, we can have a sparse matrix. Large input vectors will result in a great deal of weights, which will increase the amount of training computation needed. Word embeddings offer an answer to these issues.

GloVe

Researchers Jeffrey Pennington, Richard Socher, and Chris Manning from the Stanford NLP Group have released a programme called GloVe (Global Vectors for Word Representation) for learning continuous-space vector representations of words.

These word vectors have been demonstrated to have intriguing semantic and syntactic regularities. For instance, we discover that the following statements are true for the related word.

King - man + woman=queen
Brought - bring + seek=sought



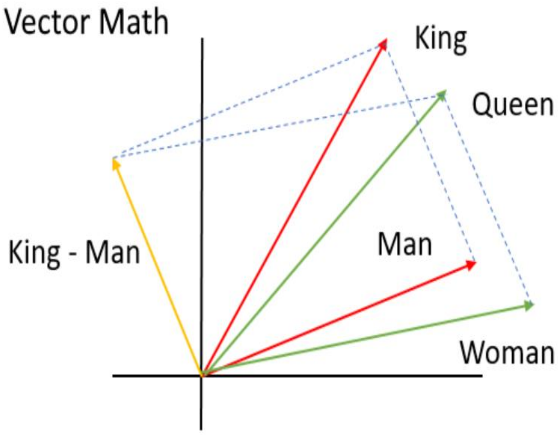


Figure 3.11: GloVe preserving semantic and syntactic regularities

Figure 3.12: vector space of words

Glove is based on a word context matrix factorization approach. It initially creates a sizable matrix of information on how frequently different words occur in different contexts across a big corpus of literature.

Understanding the two major methods on which GloVe was constructed is essential to understanding how it functions.

1. Global matrix factorization: Global matrix factorization, used in natural language processing, is a method for reducing big term frequency matrices by using matrix factorization from linear algebra. Such matrices typically show where words appear or don't appear in between text.

2. Local context window: For local context windows, there are CBOW and Skip-Gram approaches.

Glove is a method of word vector representation where using the corpus's worldwide averaged word-word co-occurrence data, training is conducted. In other words, it leverages context to comprehend and produce word representations, just like word2vec. It is strongly advised that you read the scientific article describing the procedure, Glove: Global Vectors for Word Representation, as it quickly outlines some of Word2Vec's and LSA drawbacks before detailing their own strategy.

The paper's author notes that ratios of these co-occurrence probability were more beneficial to learn than the raw occurrence probabilities alone. The embeddings are designed in such a way that the log of the incidence with which any two words will occur together is equal to a vector pair's dot product.

For instance, if in the document corpus the terms "cat" and "dog" appear 20 times in a span of 10 words while being used in the same context, then-

$$\text{Vector(c). Vector(d)} = \log (10)$$

In order to provide a more comprehensive definition, the model is forced to take the frequency distribution of surrounding terms into account.

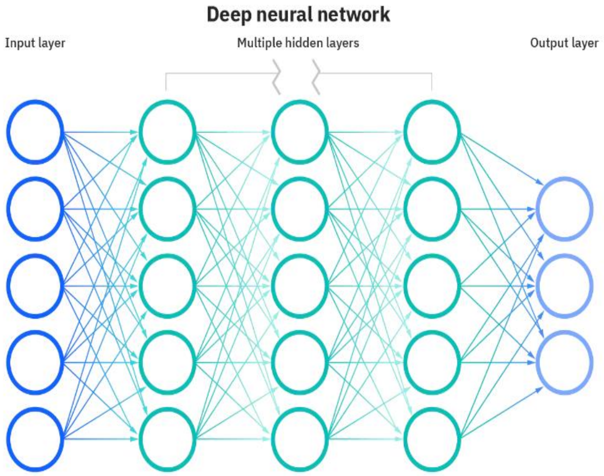
3.6 Neural Networks

In order to fine-tune their parameters, neural networks conduct calculations like addition and matrix multiplication on the numerical vectors or matrices that make up their input. A neural network's performance is influenced by a number of variables, initializing its weights and biases as well as the optimizer, loss function, and each layer's activation function.

Utilizing A layer's activation functions, weights, and biases are created to produce the output for the layer above it given a certain input. The final layer is the output layer, which displays the outcomes. The difference between each input's computed and desired results is calculated using a loss function. A neural network employs the optimizer to lower the error between the computed and intended results. An optimization technique is used to train the weights once they are randomly initialized. You can pick from a wide selection of activation functions, loss functions and optimizers.

False news detection can be viewed as a task involving binary categorization. The outcomes show whether an item of information is true or false.

Figure 3.13: Neural Network



Convolutional Neural Network (CNN)

It is a feed-forward artificial neural network. It uses multilayer perceptron's which are designed to require minimal pre-processing. When applied to NLP for text classification the sentences in the document file are represented as a square matrix, each row of which corresponds to a vector which represents a word (word embedding). Then convolution is performed on the matrix using linear filters.

A Convolutional Neural Network (CNN) provides outcomes that can be included into further training phases via matrix multiplication. This method's name is convolution Words in a sentence or a news item are represented as word vectors in the context of NLP. The training of a CNN is then done using these word vectors. By choosing a number of filters and size of the kernel, the training is done. CNN may have several dimensions.

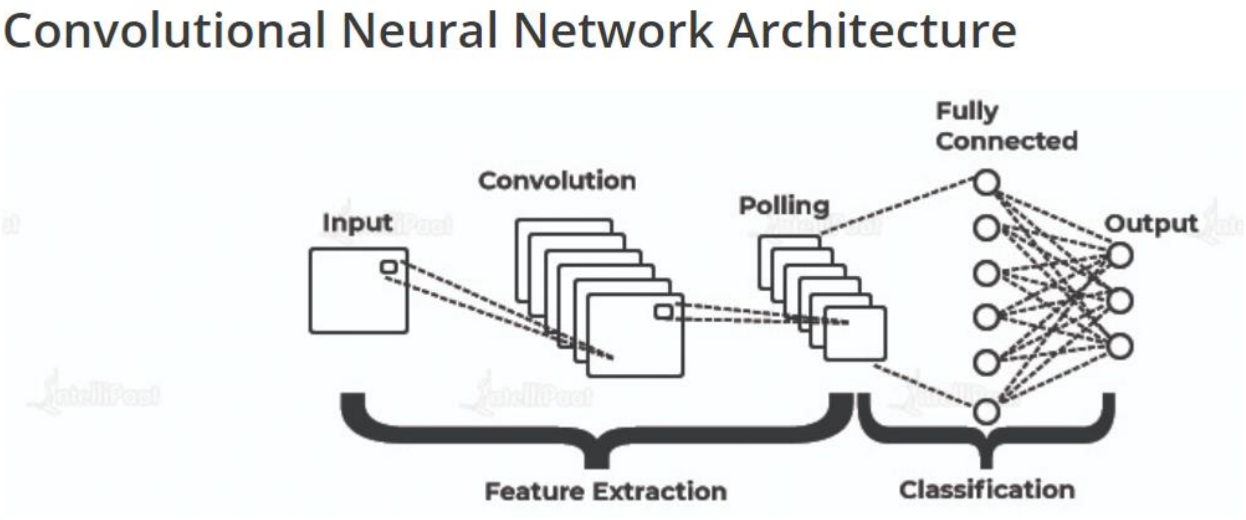


FIGURE 3.14: CNN

Typically, an unsupervised one-dimensional CNN (Conv1D) is used for text classification or NLP. Word vectors are represented as one-dimensional arrays called Conv_1D. In order to build an output that can be saved in an output array, training data is iterated over by a fixed-size window, which multiplies the input by the filter weights at each step. A feature map or output filter of the data is represented by this output array. The input training data is used to identify a feature in this manner. This procedure is illustrative.in Fig 3.15.

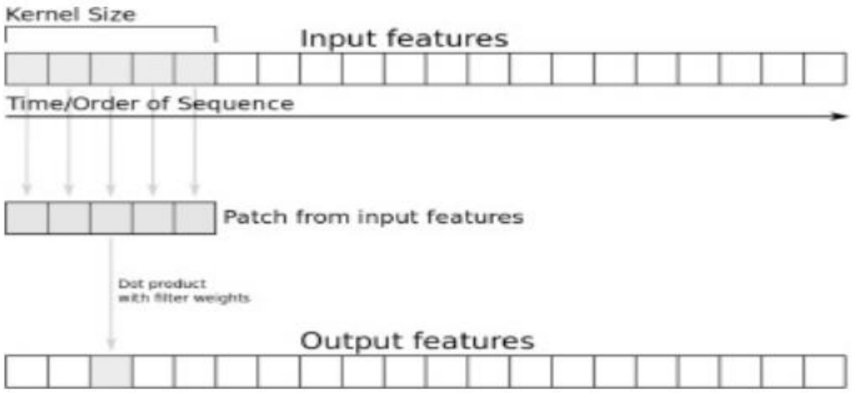


Figure 3.15: 1-D convolution operation

The number of filters and kernel size are specified, and the number of feature maps is determined by the number of filters. CNN will be able to learn local features directly from the practice set in this way.

There can be several layers in CNN. They can be of three types:

i) Convolutional

ii) Max Pooling

iii) Fully-connected

Assume that an N-N square neuron layer comes after the number of filters and kernel size are specified, and the number of feature maps is determined by the number of filters. CNN will be able to learn local features directly from the practice set in this way.

$$x_{ij}^{\ell} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \omega_{ab} \, y_{(i+a)(j+b)}^{\ell-1}.$$

To summarize

1. CNN layer accepts an input volume with the dimensions Winput x Hinput xDinput (Winput = Hinput is a common occurrence because inputs are normally squares).

2. Calls for four parameters

- The number of filters k.
- The Kernel size used for convolution, called the receptive field size F, is always square and results in a FxF kernel.
- The stride S
- P (zero padding)

$$\text{Woutput} = ((\text{Winput} - F + 2P) / S) + 1$$
$$\text{Houtput} = ((\text{Hinput} - F + 2P) / S) + 1$$
$$\text{Doutput} = K$$

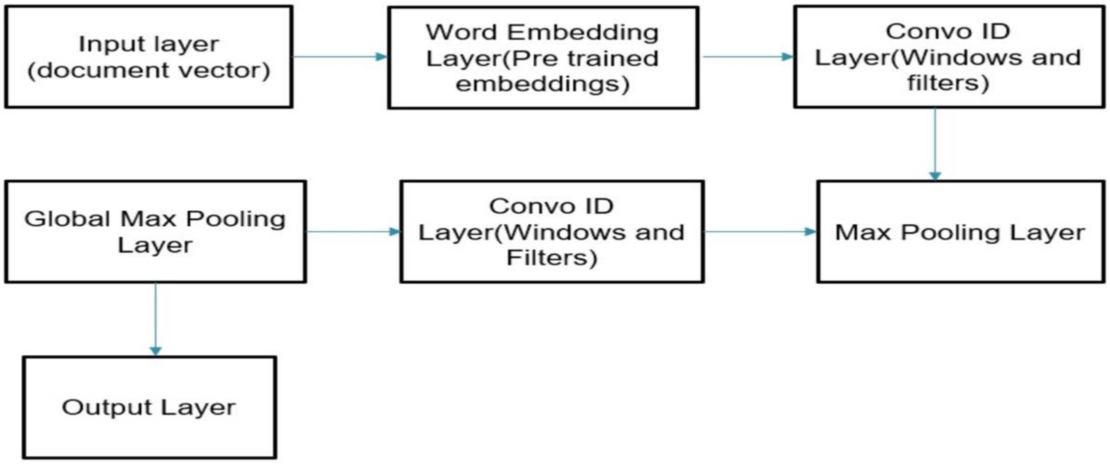
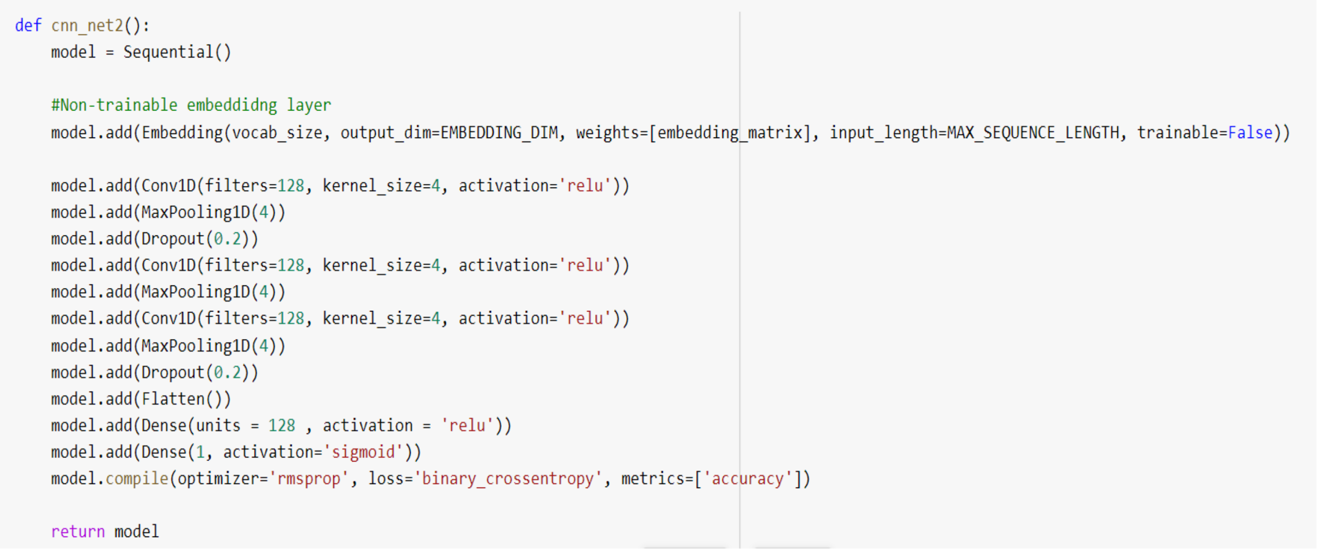

Input layer (document vector)
Word Embedding Layer(Pre trained embeddings)
Convo ID Layer(Windows and filters)
Max Pooling Layer
Convo ID Layer(Windows and Filters)
Global Max Pooling Layer
Output Layer

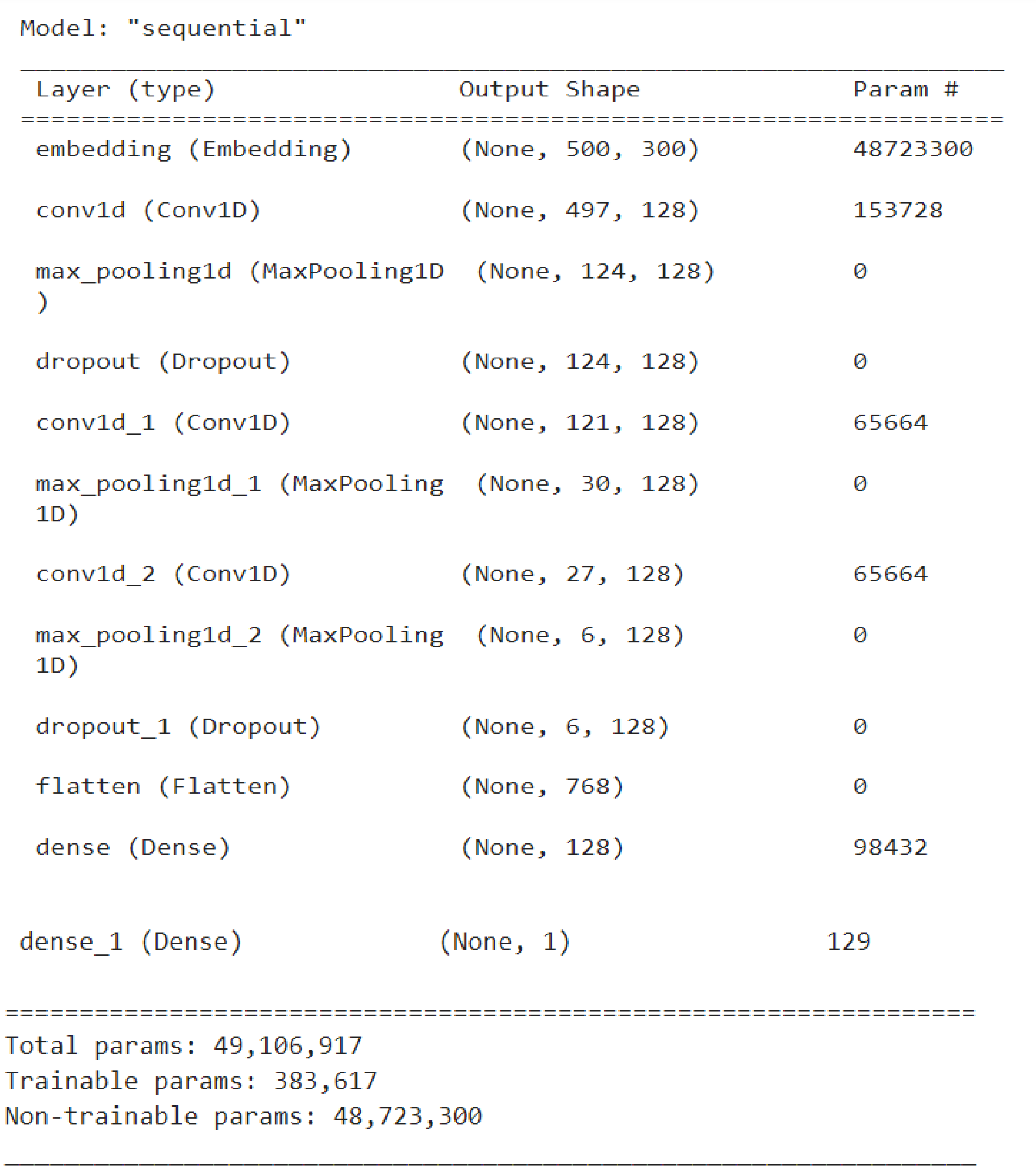
Figure 3.16: CNN Layers

Model Implementation



```
def cnn_net2():
    model = Sequential()

    #Non-trainable embeddidng layer
    model.add(Embedding(vocab_size, output_dim=EMBEDDING_DIM, weights=[embedding_matrix], input_length=MAX_SEQUENCE_LENGTH, trainable=False))

    model.add(Conv1D(filters=128, kernel_size=4, activation='relu'))
    model.add(MaxPooling1D(4))
    model.add(Dropout(0.2))
    model.add(Conv1D(filters=128, kernel_size=4, activation='relu'))
    model.add(MaxPooling1D(4))
    model.add(Conv1D(filters=128, kernel_size=4, activation='relu'))
    model.add(MaxPooling1D(4))
    model.add(Dropout(0.2))
    model.add(Flatten())
    model.add(Dense(units = 128 , activation = 'relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['accuracy'])

    return model
```

Figure 3.17 CNN Model



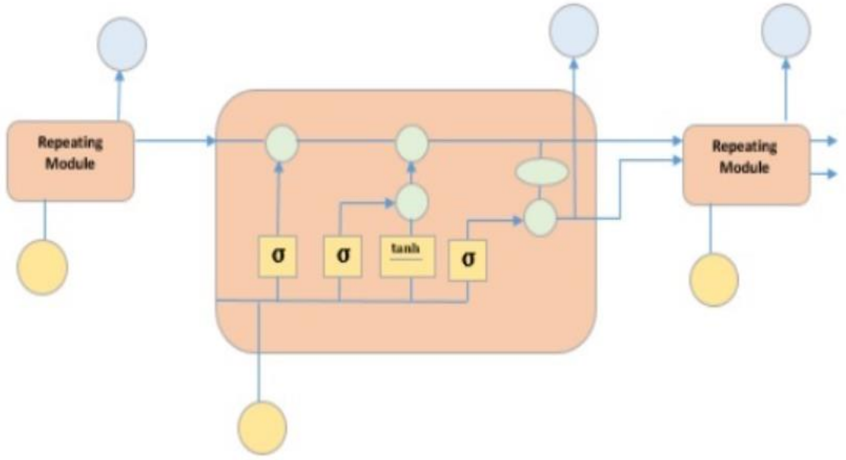
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 500, 300)          48723300

 conv1d (Conv1D)             (None, 497, 128)          153728

 max_pooling1d (MaxPooling1D  (None, 124, 128)         0
 )

 dropout (Dropout)           (None, 124, 128)          0

 conv1d_1 (Conv1D)           (None, 121, 128)          65664

 max_pooling1d_1 (MaxPooling  (None, 30, 128)          0
 1D)

 conv1d_2 (Conv1D)           (None, 27, 128)           65664

 max_pooling1d_2 (MaxPooling  (None, 6, 128)           0
 1D)

 dropout_1 (Dropout)         (None, 6, 128)            0

 flatten (Flatten)           (None, 768)               0

 dense (Dense)               (None, 128)               98432


dense_1 (Dense)              (None, 1)                 129

=================================================================
Total params: 49,106,917
Trainable params: 383,617
Non-trainable params: 48,723,300
_________________________________________________________________
```

Figure 3.18 CNN Model Summary

Long-Short Term Memory (LSTM)

The Long-Short Term Memory network is what it stands for. The RNN (Recurrent Neural Network) variant known as LSTM is specifically created to address long-term dependency issues. Because it has internal memory that can be used to process arbitrary input sequences and can tolerate arbitrary input/output length, LSTM is excellent for classifying text and audio. In LSTM, the analysis of sequential data is possible because some layer outputs are fed back into the inputs of a prior layer.

Figure 3.19: LSTM



As shown in FIGURE 3.19, LSTMs comprise both LTM and STM and make use of the concept of gates for making the calculations simple and effective. [7]

1. Forget Gate: After entering the Forget Gate, LTM forgets useless information.
2. Learn Gate: In order to apply recent knowledge from STM to an event (or current input), STM and the event are integrated.
3. Remember Gate: Remember gate, which serves as a modernized LTM, combines LTM data that hasn't been forgotten with STM and Event data.
4. Use Gate: This gate serves as an updated STM and forecasts the outcome of the event using LTM, STM, and Event.

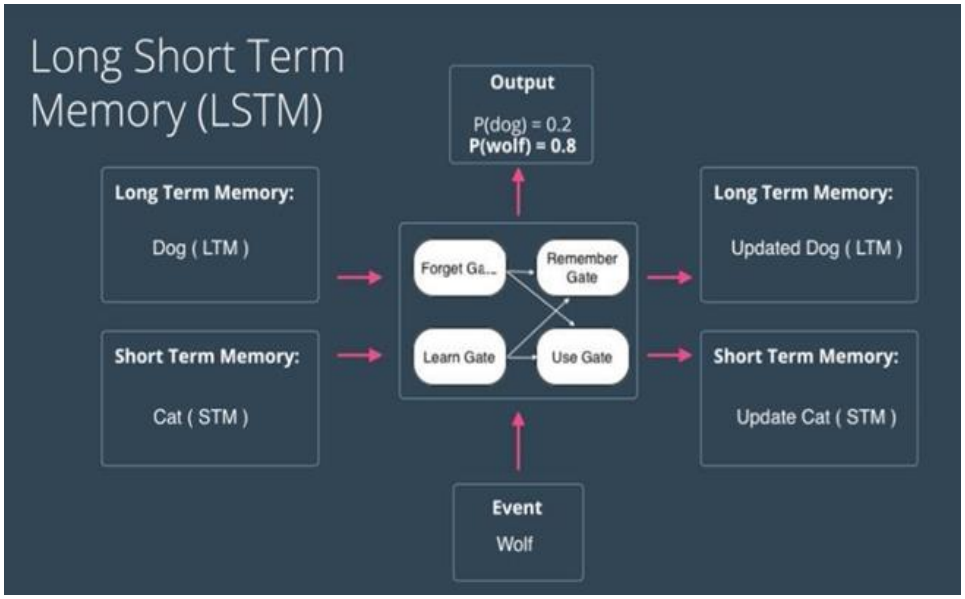


Figure 3.20 LSTM gates

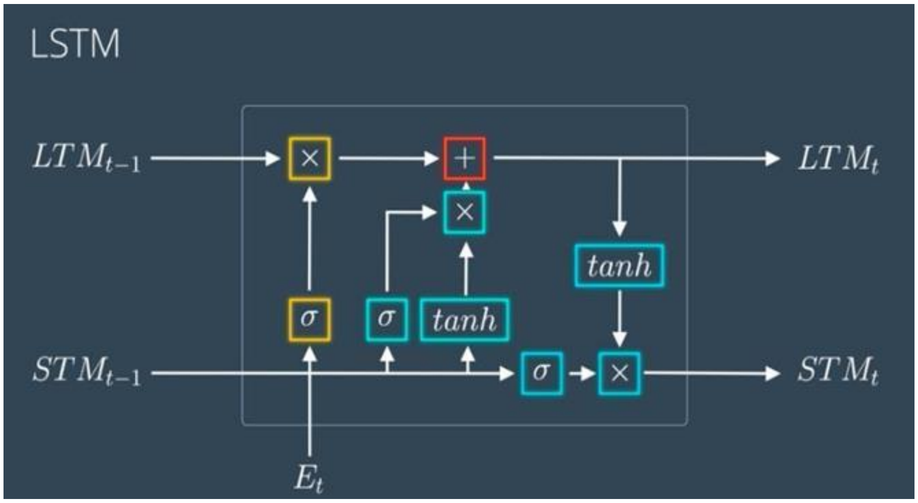


Figure 3.21: LSTM structure

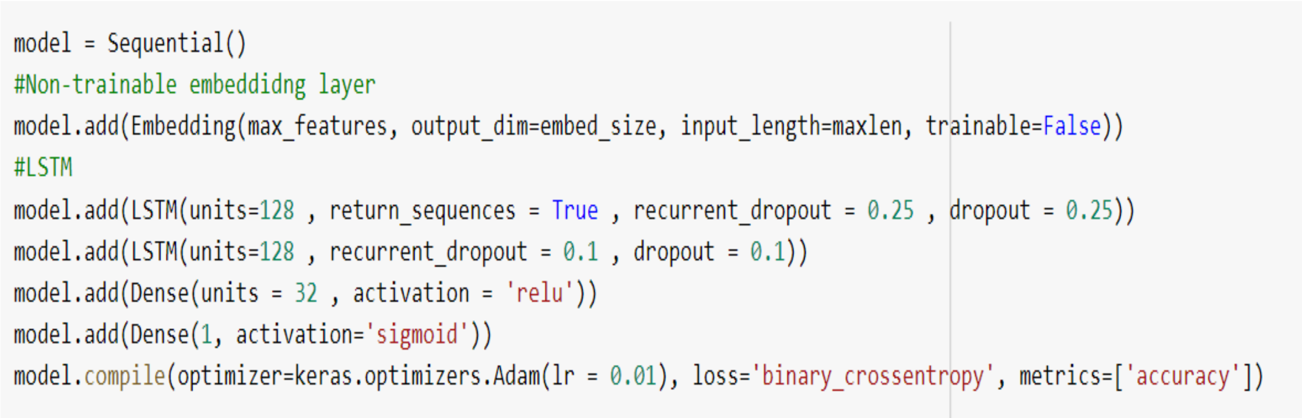
In FIGURE 3.21, we can see the architecture on how LSTM works.

Usage of LSTMs

Although it resolves (or eliminates) the Vanishing Gradient problem (too-small weights that under-fit the model), it still has to deal with the Exploding Gradient problem (weights become too large that over-fits the model).

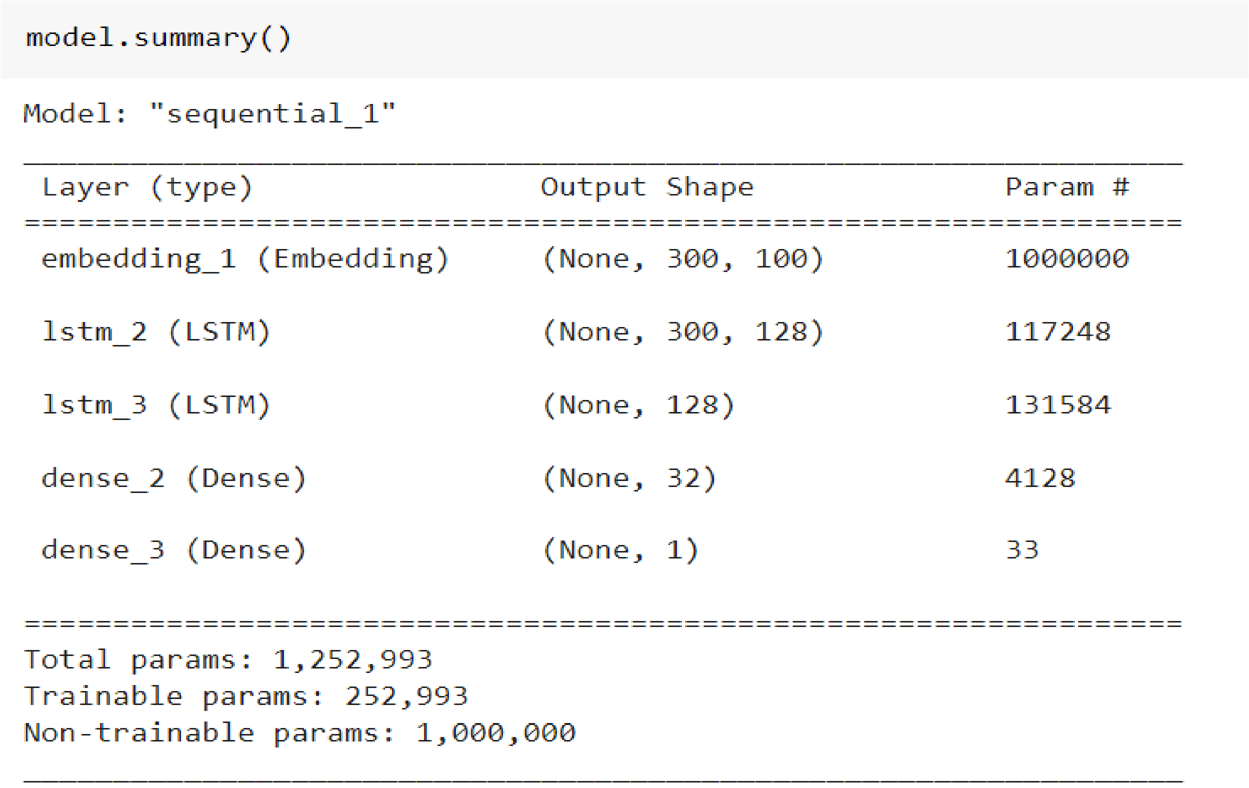
LSTMs are frequently employed in tasks like language generation, voice recognition, image OCR models, object detection, etc. because they take care of the long-term dependencies.

Model Implementation



```
model = Sequential()
#Non-trainable embeddidng layer
model.add(Embedding(max_features, output_dim=embed_size, input_length=maxlen, trainable=False))
#LSTM
model.add(LSTM(units=128 , return_sequences = True , recurrent_dropout = 0.25 , dropout = 0.25))
model.add(LSTM(units=128 , recurrent_dropout = 0.1 , dropout = 0.1))
model.add(Dense(units = 32 , activation = 'relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer=keras.optimizers.Adam(lr = 0.01), loss='binary_crossentropy', metrics=['accuracy'])
```

Figure 3.22: LSTM Model



```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 300, 100)          1000000

 lstm_2 (LSTM)               (None, 300, 128)          117248

 lstm_3 (LSTM)               (None, 128)               131584

 dense_2 (Dense)             (None, 32)                4128

 dense_3 (Dense)             (None, 1)                 33

=================================================================
Total params: 1,252,993
Trainable params: 252,993
Non-trainable params: 1,000,000
_________________________________________________________________
```

Figure 3.23: LSTM Model Summary

Hybrid CNN-LSTM model

The suggested model makes use of the long-term dependency learning capabilities of the LSTM and the local feature extraction capabilities of the CNN. The input vectors are first processed by a Conv1D that is a convolution layer, which is then utilized to extract the text-level local features. The Following LSTM units/cells is an RNN layer uses the feature maps created by the CNN layer as its input. CNN’s local features are used by the RNN layer to determine if news stories are fake or real by learning.

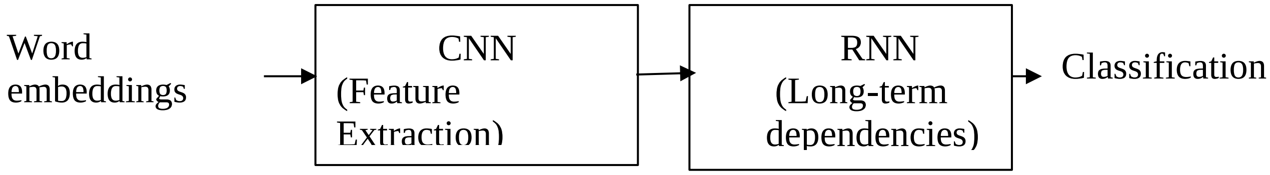


Figure 3.24: Hybrid Model

Due to their ability to discern between local and sequential input data properties, A number of classification and regression applications have seen success with CNN-RNN. For instance, they are used for the recognition of sign language from video streams and to detect emotions (Kollias and Zafeiriou, 2020; (2018) Masood et al. These applications make use of the RNN's and CNN's capacity to learn consecutive characteristics and scene components. RNN can capture long-term relationships between text entities and essential attributes in the context of NLP tasks and can learn temporal and context features from text. found utilizing CNN's capability to handle spatial relations (Zhang, Chen, Huang, 2018, Zhou, Sun, Liu, & Lau).

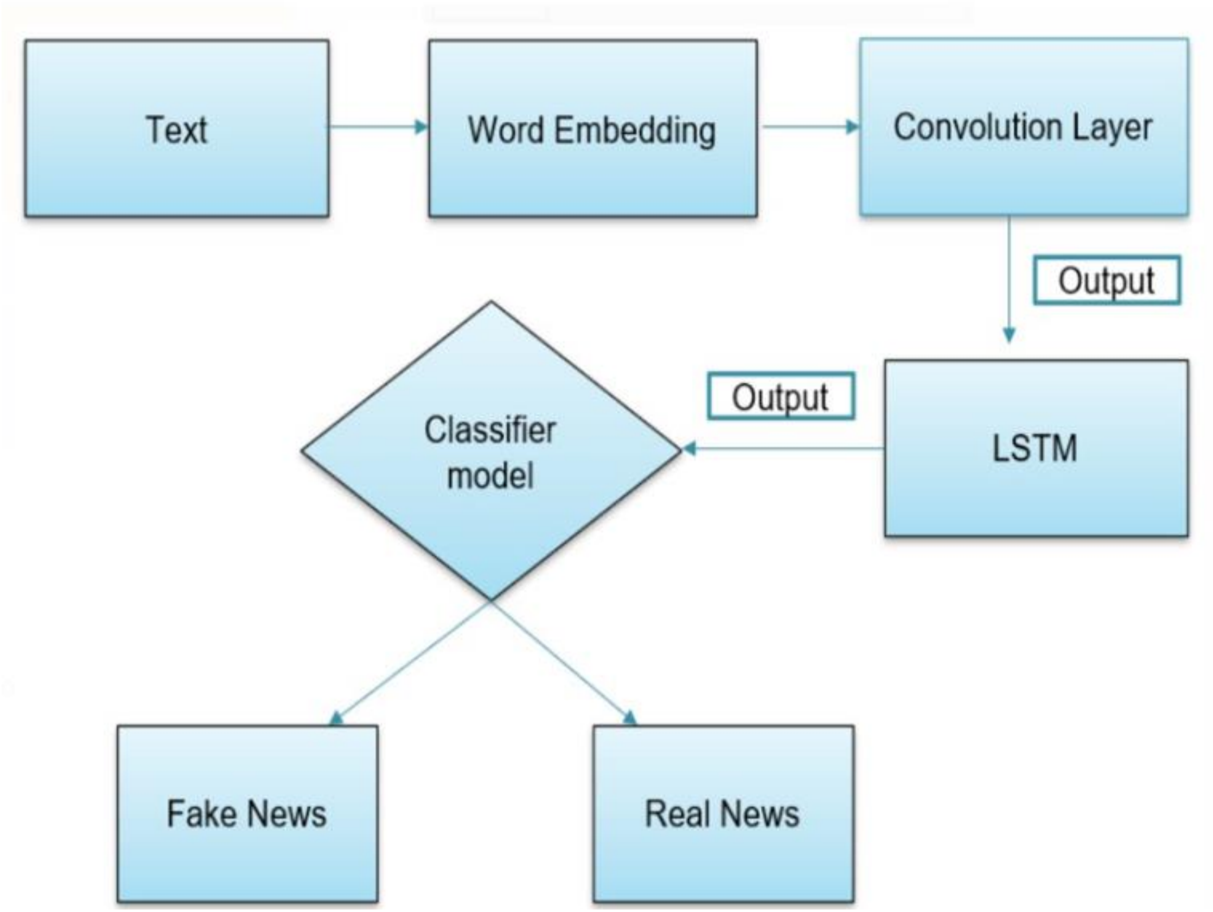
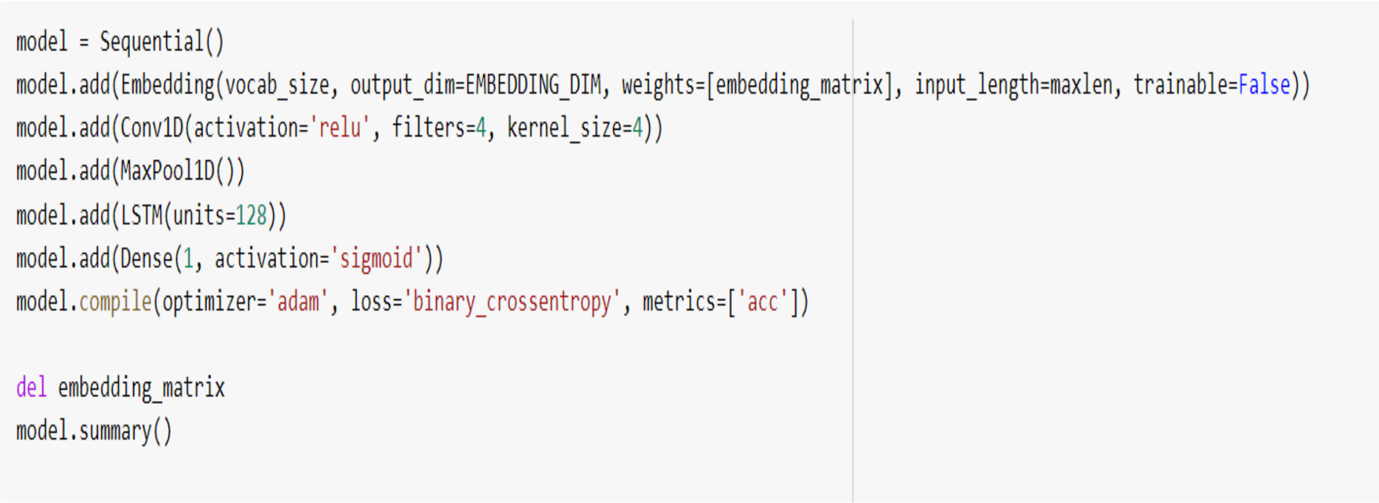
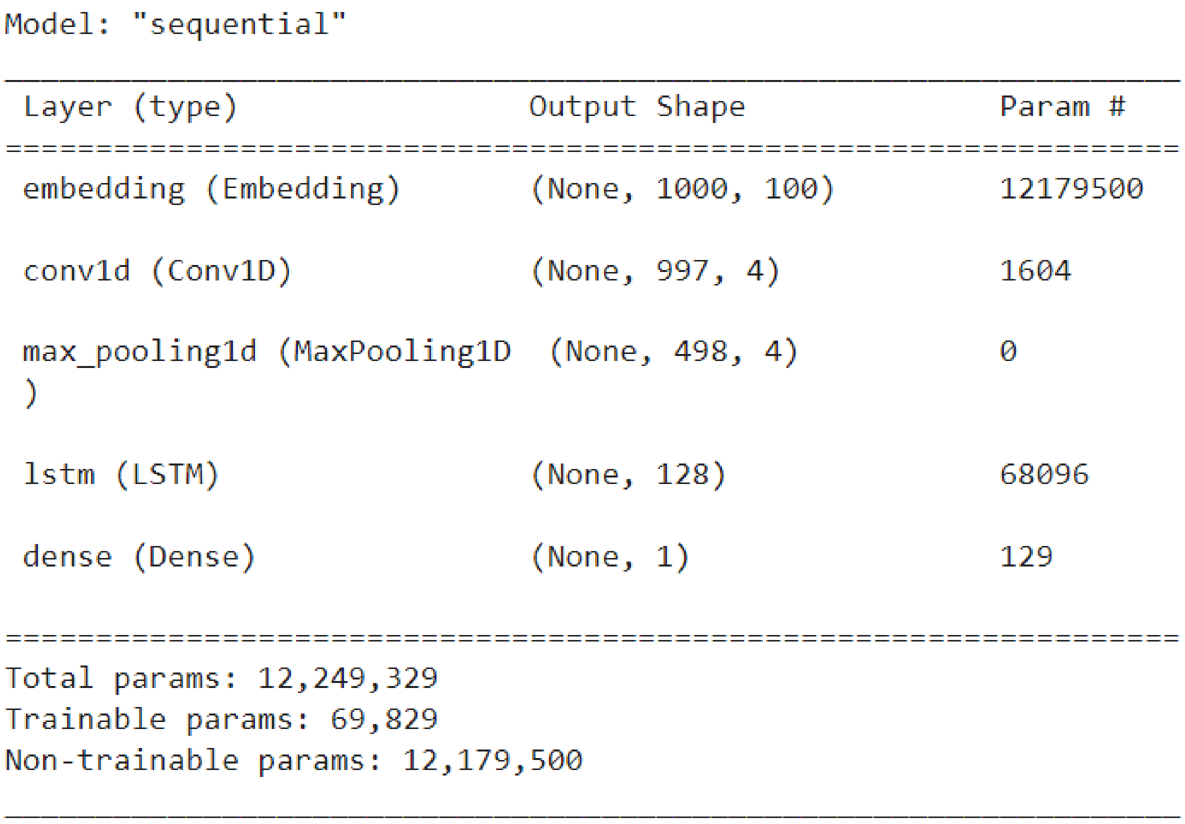


Figure 3.25: Flow of Hybrid (CNN +LSTM)



```
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()
```

Figure 3.26: Model of Hybrid (CNN +LSTM)



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding (Embedding)       (None, 1000, 100)         12179500  
                                                                 
 conv1d (Conv1D)             (None, 997, 4)            1604      
                                                                 
 max_pooling1d (MaxPooling1D  (None, 498, 4)           0         
 )                                                               
                                                                 
 lstm (LSTM)                 (None, 128)               68096     
                                                                 
 dense (Dense)               (None, 1)                 129       
                                                                 
=================================================================
Total params: 12,249,329
Trainable params: 69,829
Non-trainable params: 12,179,500
_________________________________________________________________
```

Figure 3.27: Model of Hybrid (CNN +LSTM) Summary

FLOW OF THE PROJECT

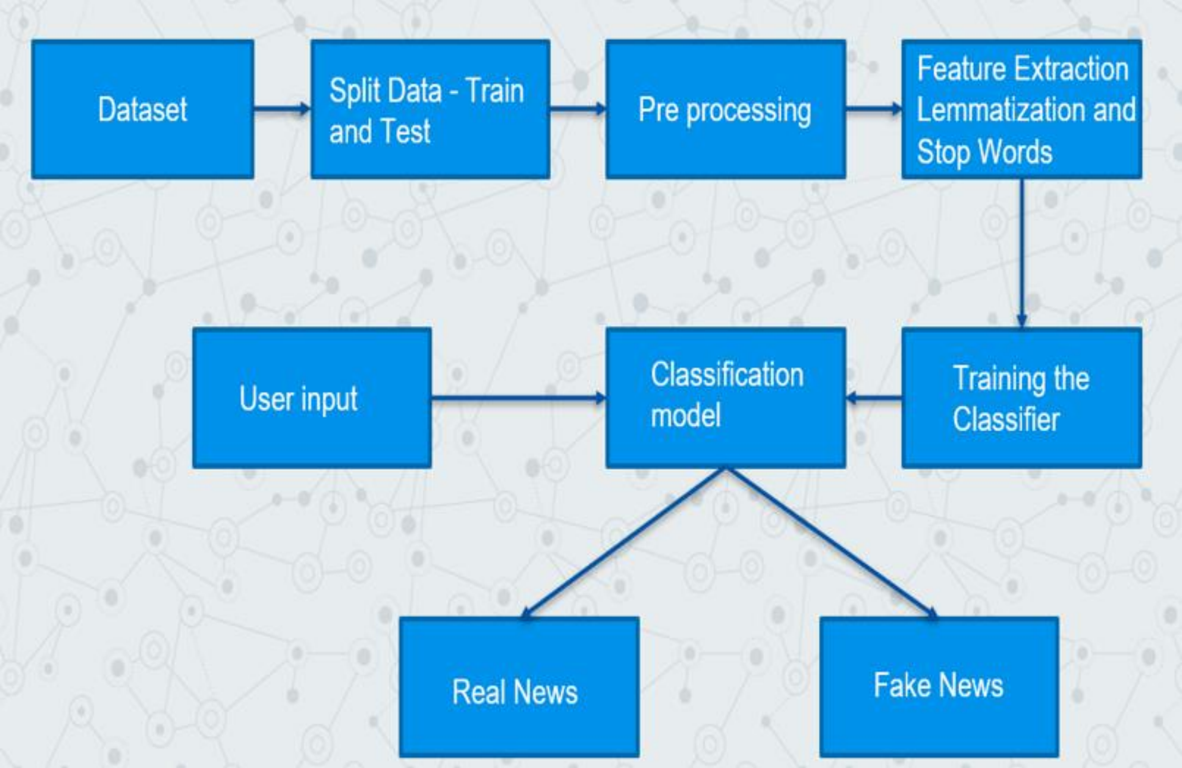


Figure 3.28: Flow of Project

3.7 Mathematical

1. **ReLU function**

   The rectified linear activation function is a straightforward calculation that gives an immediate response of the value entered or 0.0 if the input is 0.0 or less.

$$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

Figure 3.29: Equation for ReLU function

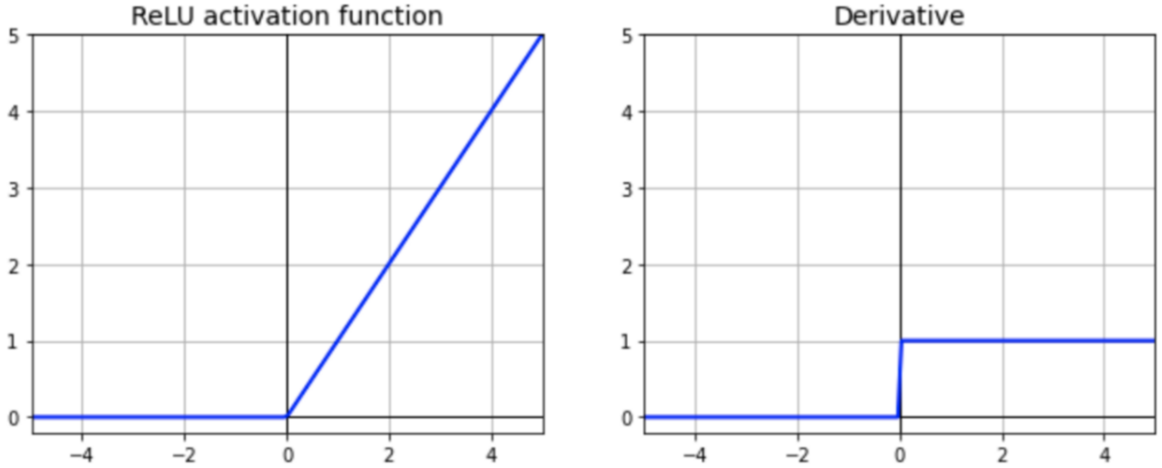


Figure 3.30: Plot of ReLU activation function and its derivative

Rectified Linear Activation Function Benefits

- To account for non-linearities, the ReLU function is divided graphically into two linear halves. If the slope of a function changes, it isn't linear. Negative inputs have 0 slope while all positive inputs have 1 slope.
- This function helps computes quite quickly.

2. **Sigmoid Function**

A mathematical function called the sigmoid function converts any input value to a number between 0 and 1. It is frequently used in machine learning as an activation function to provide nonlinearity to the model, particularly in logistic regression and neural networks. It is said that the sigmoid function is:

$$y(x) = 1 / (1 + e^{(-x)})$$

where x is the input value and e are Euler's number, a mathematical constant that is roughly equivalent to 2.71828. Since y(x) always has an output value between 0 and 1, it can be used to represent probabilities or binary classifications.

When x is a negative number, the sigmoid function moves closer to zero. The sigmoid function approaches 1 when x is positive. The sigmoid function produces 0.5 when x is 0.

The sigmoid function is advantageous in machine learning because it possesses a number of favorable characteristics. Since it is differentiable, any point can be used to calculate its derivative. This qualifies it for optimization techniques like gradient descent. It always increases or always drops as x increases since it is monotonic.

The sigmoid function, however, is prone to saturation, which implies that for extremely high or extremely low values of x, the function's gradient becomes extremely small. Due to slow convergence during training, the model may be unable to recognize complex patterns in the data. Alternative activation functions, like the Rectified Linear Unit (ReLU), have been created to address this problem.

3. Adam Optimizer

Adam (Adaptive Moment Estimation) is a well-liked stochastic gradient descent (SGD) optimization technique used in machine learning. AdaGrad and RMSProp, two further optimization methods, are combined to create it.

The first and second moments, which are the past gradients and past squared gradients, respectively, are maintained by the Adam optimizer. Based on the size of the gradient and the magnitude of the second moment of the gradient, the algorithm adjusts the learning rate for each parameter. As a result, the algorithm converges more quickly and consistently than conventional optimization algorithms.

Based on the following equations, the Adam optimizer modifies the model's parameters:

m_t = beta1 * m_(t-1) + (1 - beta1) * g_t

v_t = beta2 * v_(t-1) + (1 - beta2) * (g_t ^ 2)

m_hat = m_t / (1 - beta1^t)

v_hat = v_t / (1 - beta2^t)

theta_t = theta_(t-1) - alpha * m_hat / (sqrt(v_hat) + epsilon)

where g_t is the gradient at time step t, m_t and v_t is the first and second moments, m_hat and v_hat are the bias-corrected estimates of the first and second moments, theta_t is the updated parameter at time step t, alpha is the learning rate, beta1 and beta2 are the exponential decay rates for the first and second moments, and epsilon is a small value added to the denominator to avoid division by zero.

The Adam optimizer is superior to conventional optimization techniques in a number of ways. It is suitable for issues involving huge datasets and high-dimensional parameter spaces since it is computationally efficient, memory-intensive, and memory-constrained. Additionally, it offers adaptive learning rates, which can help the model's convergence and generalization. To attain the best performance, it is crucial to precisely set the optimizer's hyperparameters, including the learning rate, decay rates, and epsilon.

4. RMSProp

The stochastic gradient descent (SGD) optimization approach used in machine learning is called RMSProp (Root Mean Square Propagation). The learning rate for each parameter is adjusted depending on the squared gradient's historical average in this variation of the gradient descent technique.

The RMSProp algorithm employs this average, which has an exponential decreasing trend, to normalize the gradient at each iteration. This smoothes the optimization process and lessens the influence of unpredictable or noisy gradients.

Based on the following equations, the RMSProp optimizer modifies the model's parameters:

g_t = gradient at time t

E[g^2] _t = beta * E[g^2] _(t-1) + (1 - beta) * g_t^2

theta_t = theta_(t-1) - alpha / sqrt(E[g^2] _t + epsilon) * g_t

where g_t is the gradient at time step t, E[g^2] _t is the moving average of the squared gradient at time step t, theta_t is the updated parameter at time step t, alpha is the learning rate, beta is the decay rate of the moving average, and epsilon is a small value added to the denominator to avoid division by zero.

When compared to conventional optimization algorithms, the RMSProp optimizer has a number of benefits. It is suitable for issues involving huge datasets and high-dimensional parameter spaces since it is computationally efficient, memory-intensive, and memory-constrained. Additionally, it offers adaptive learning rates, which can help the model's convergence and generalization. To attain the best performance, it is crucial to precisely set the optimizer's hyperparameters, including the learning rate, decay rate, and epsilon.

5. **Binary_Crossentropy**

A loss function that is frequently used in machine learning for binary classification tasks is binary crossentropy, also referred to as log loss. It calculates the discrepancy between the actual binary label distribution and the expected probability distribution.

The binary crossentropy formula is:

$$L\ (y, y') = -\ [y * \log(y') + (1 - y) * \log\ (1 - y')]$$

where y' is the anticipated probability of the positive class, or the likelihood that the label is 1, and y represents the true binary label (0 or 1).

When the anticipated probability distribution coincides with the actual binary label distribution, the binary crossentropy loss function is minimized. Because it is simple to calculate, offers a continuous measure of the loss, and works well with backpropagation-based optimization algorithms like stochastic gradient descent, it is a popular choice for binary classification problems.

In actuality, the sigmoid activation function, which converts the output of the model to a probability between 0 and 1, is frequently used with binary crossentropy. For binary classification issues, the sigmoid function and binary crossentropy loss function work well together.

Chapter 04: Experiments and Results Analysis

The following variables were used to evaluate the accuracy of the predictions:

- **Accuracy**: This evaluates how accurately the model's forecasts were made overall. It is the ratio of samples that were correctly categorized to all of the samples in the dataset.

  Accuracy = (Number of correctly classified samples) / (Total number of samples)

- **Precision:** This calculates the ratio of true positives (positive samples that were accurately predicted to be positive) to the total number of samples that were projected to be positive.

  Precision = (Number of true positives) / (Number of true positives + Number of false positives)

- **Recall:** This measures the ratio of genuine positive samples in the dataset to the overall number of true positives.

  Recall = (Number of true positives) / (Number of true positives + Number of false negatives)

- **F1 Score:** A single metric that balances precision and recall is provided by the F1 score, which is the harmonic mean of the two metrics.

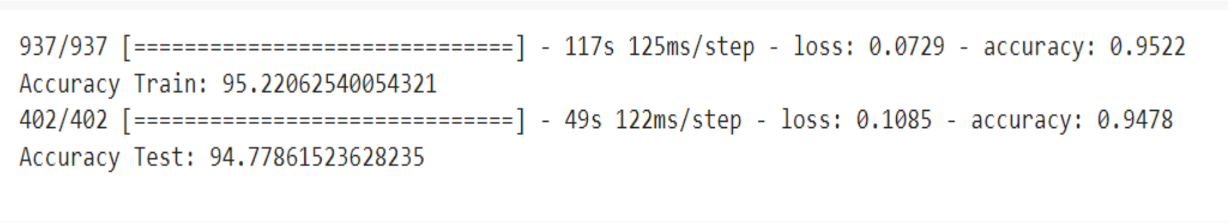
  F1 score = 2 * (Precision * Recall) / (Precision + Recall)

- **Learning curves:** These are charts that display the model's accuracy or loss with time during training and validation. They can be used to determine the ideal number of training epochs as well as to diagnose underfitting or overfitting.

The model was trained by three Deep Learning Algorithms

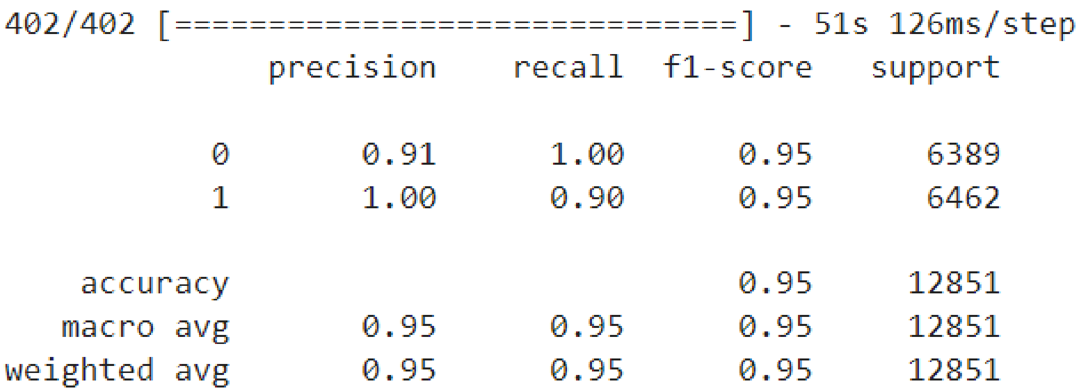
- CNN
- LSTM
- Hybrid (CNN+LSTM)

CNN Model Result



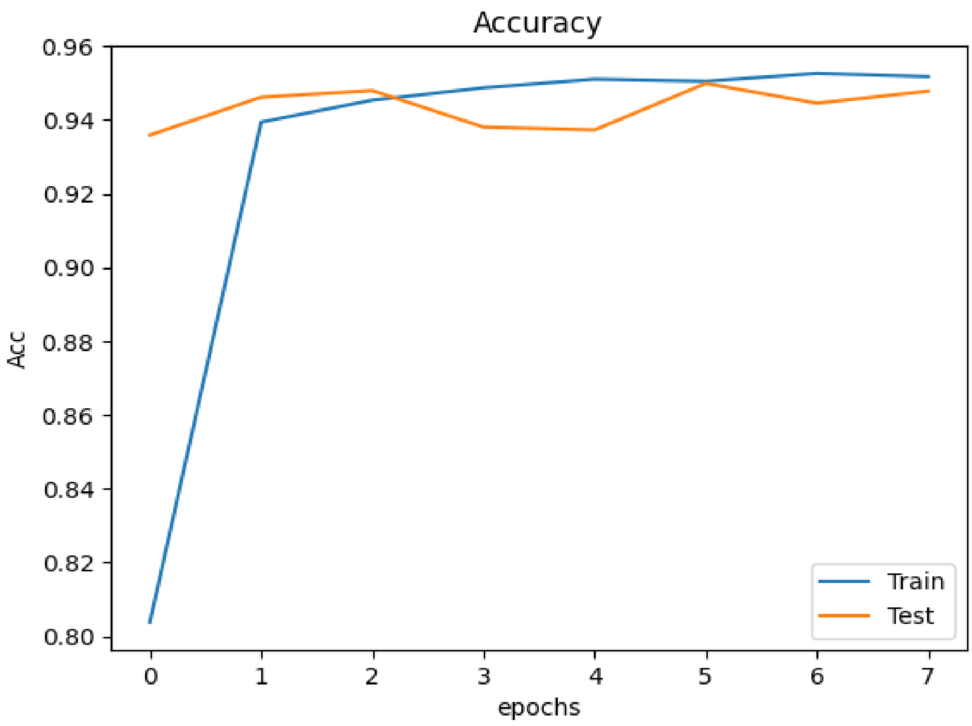
```
937/937 [==============================] - 117s 125ms/step - loss: 0.0729 - accuracy: 0.9522
Accuracy Train: 95.22062540054321
402/402 [==============================] - 49s 122ms/step - loss: 0.1085 - accuracy: 0.9478
Accuracy Test: 94.77861523628235
```

Figure 4.1: Accuracy of the training and testing data

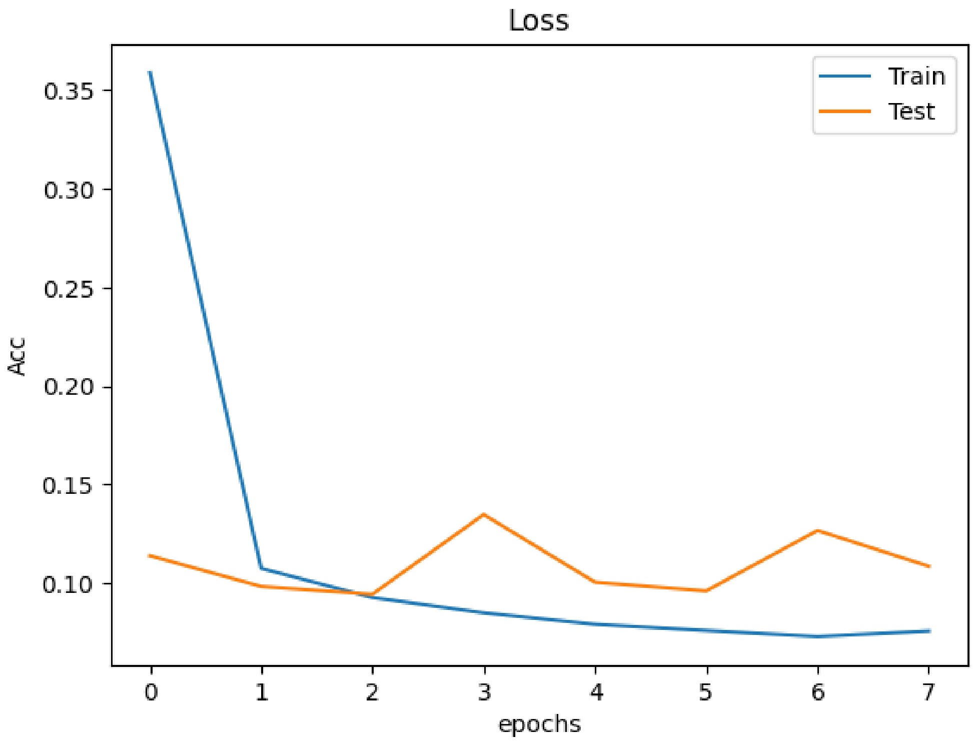


```
402/402 [==============================] - 51s 126ms/step
              precision    recall  f1-score   support

           0       0.91      1.00      0.95      6389
           1       1.00      0.90      0.95      6462

    accuracy                           0.95     12851
   macro avg       0.95      0.95      0.95     12851
weighted avg       0.95      0.95      0.95     12851
```

Figure 4.2: Classification Report of CNN Model

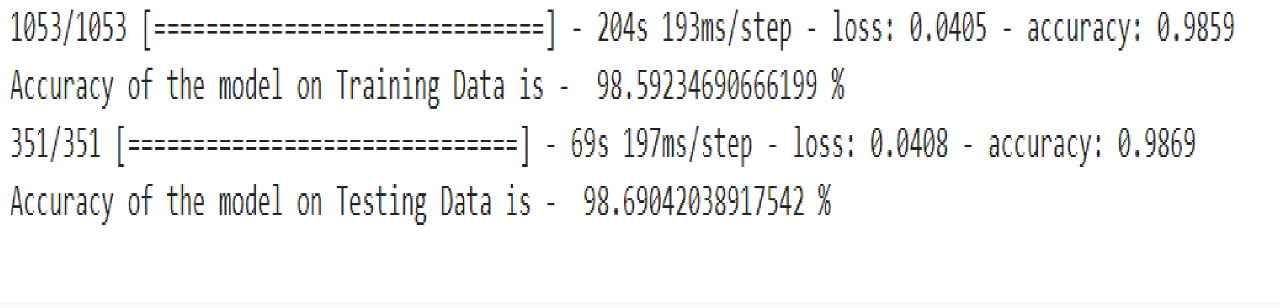


Graph 4.1: Accuracy VS Epoch Graph of CNN Model



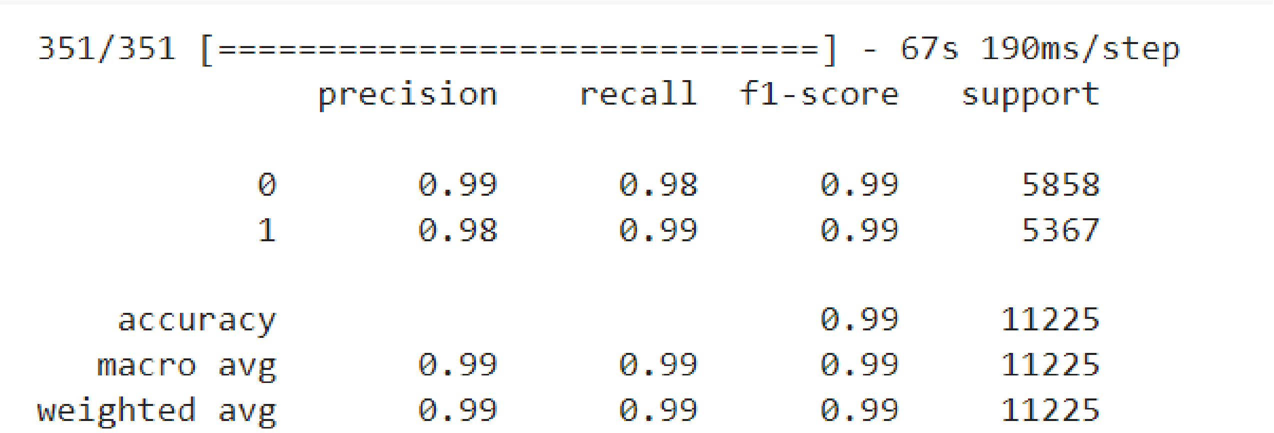
Graph 4.2: Loss VS Epoch Graph of CNN Model

LSTM Model Result



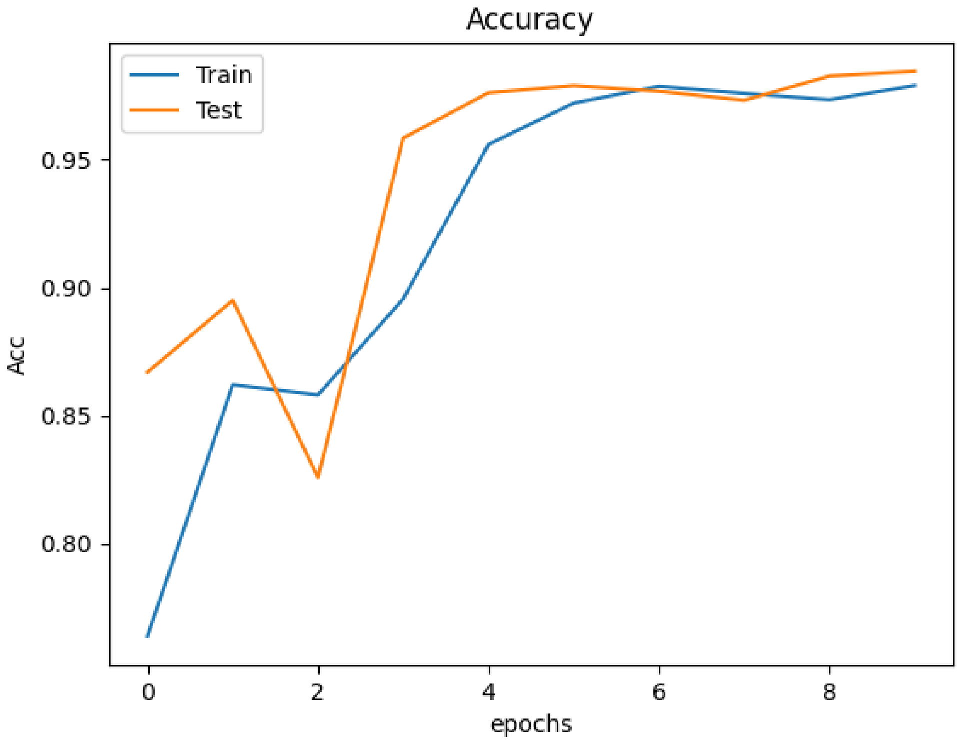
```
1053/1053 [==============================] - 204s 193ms/step - loss: 0.0405 - accuracy: 0.9859
Accuracy of the model on Training Data is -  98.59234690666199 %
351/351 [==============================] - 69s 197ms/step - loss: 0.0408 - accuracy: 0.9869
Accuracy of the model on Testing Data is -  98.69042038917542 %
```

Figure 4.3: Accuracy of the training and testing data

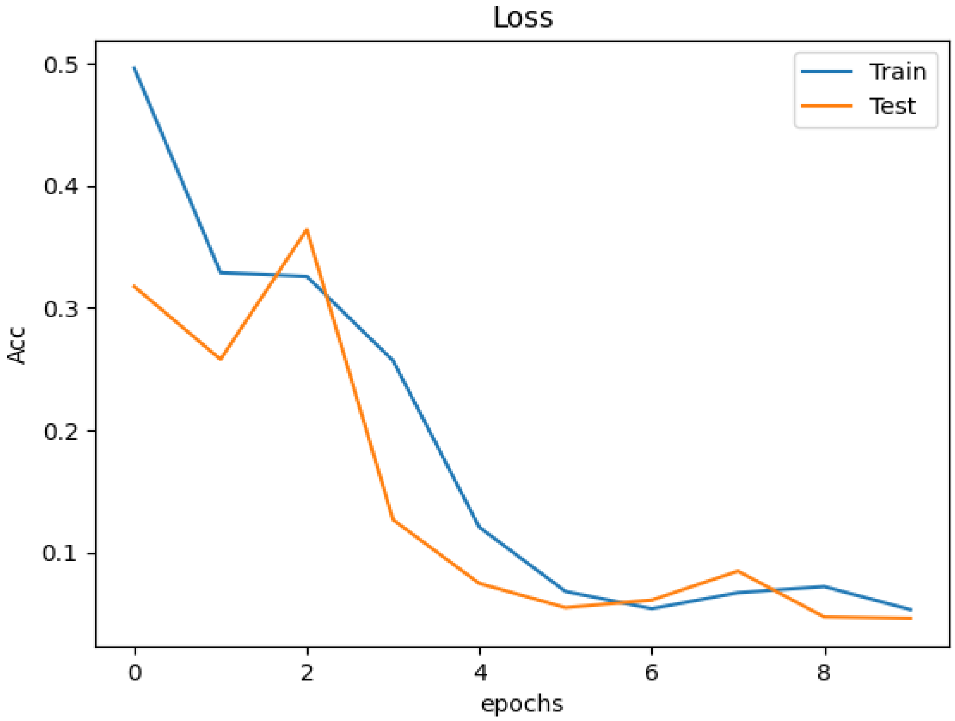


```
351/351 [==============================] - 67s 190ms/step
              precision    recall  f1-score   support

           0       0.99      0.98      0.99      5858
           1       0.98      0.99      0.99      5367

    accuracy                           0.99     11225
   macro avg       0.99      0.99      0.99     11225
weighted avg       0.99      0.99      0.99     11225
```

Figure 4.4: Classification Report of LSTM Model

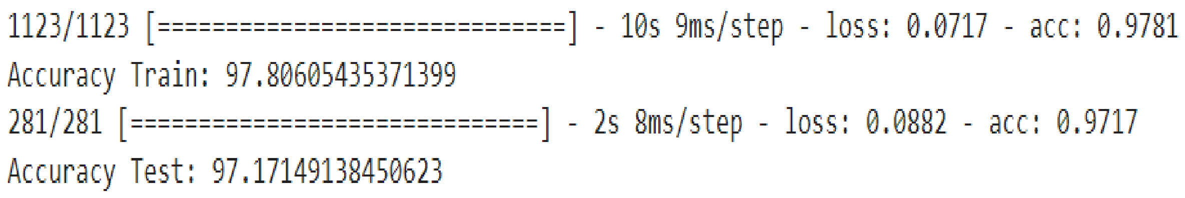


Graph 4.3: Accuracy VS Epoch Graph of LSTM Model



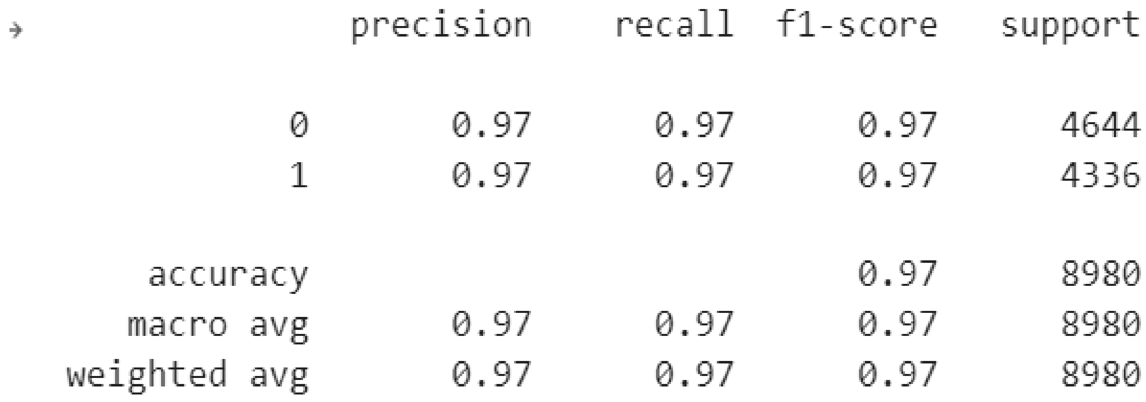
Graph 4.4: Loss VS Epoch Graph of LSTM Model

Hybrid (CNN+LSTM) Result



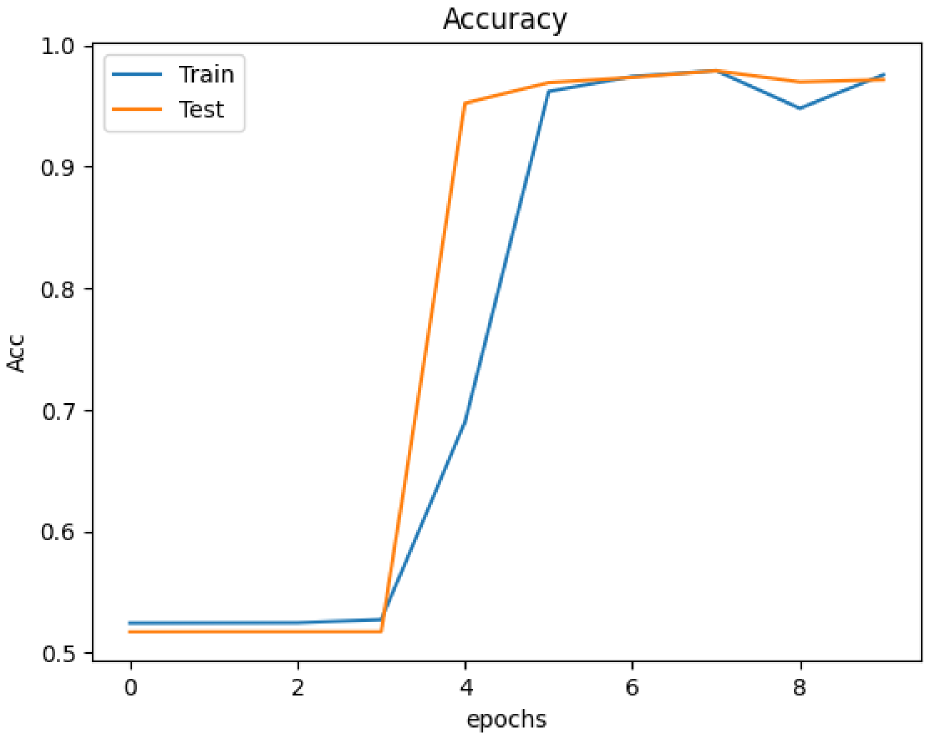
```
1123/1123 [==============================] - 10s 9ms/step - loss: 0.0717 - acc: 0.9781
Accuracy Train: 97.80605435371399
281/281 [==============================] - 2s 8ms/step - loss: 0.0882 - acc: 0.9717
Accuracy Test: 97.17149138450623
```

Figure 4.5: Accuracy of the training and testing data of Hybrid Model

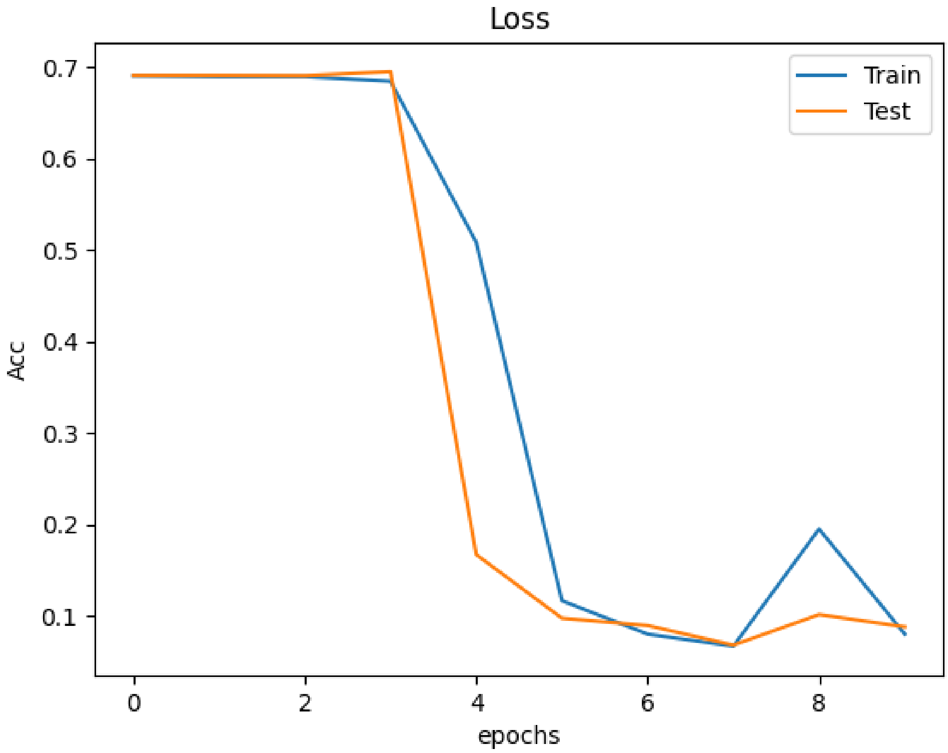


```
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      4644
           1       0.97      0.97      0.97      4336

    accuracy                           0.97      8980
   macro avg       0.97      0.97      0.97      8980
weighted avg       0.97      0.97      0.97      8980
```

Figure 4.6: Classification Report of Hybrid Model



Graph 4.5: Accuracy VS Epoch Graph of Hybrid Model



Graph 4.6: Loss VS Epoch Graph of Hybrid Model

Performance Measure comparison Using Different Algorithm

Models	Precision (%)	Recall (%)	F1 Score (%)	Accuracy (%)
CNN	95	95	95	95
LSTM	98	98	99	99
CNN+LSTM	97	97	97	97

Table 4.1: Comparison Result Table

CHAPTER 5: CONCLUSIONS

5.1 Conclusions

In conclusion, deep learning model-based trials for false news identification have produced encouraging findings. A 95% accuracy was attained by the CNN model, a 99% accuracy by the LSTM model, and a 97% accuracy by the hybrid model.

These models' high accuracy demonstrates that deep learning models are useful for identifying fake news. It is crucial to remember that accuracy does not, by itself, give a full picture of the model's performance.

Further investigation could be conducted in order to enhance the functionality of these models. To get even better results, this can entail experimenting with various architectures or fine-tuning the hyperparameters.

Overall, these studies' findings imply that CNN, LSTM, and hybrid deep learning models, in particular, have potential for accurate fake news identification.

5.2 Future Scope

Regarding the proposed hybrid model's future work, we anticipate running it on transformers similar to Bert and then conducting some tests with federated learning APIs. In light of this, we intend to post the dataset in Kaggle datasets for additional research.

Future research and development in the field of false news identification utilizing deep learning models has the potential to yield promising outcomes. Here are a few potential future avenues for this area of study:

Multi-modal data handling: At the moment, the majority of fake news detection methods are based on textual data, however fake news can also be disseminated through other media, such as photographs, videos, and audio. Future studies could examine the application of multi-modal deep learning models to the detection of fake news in various media.

Enhancing model interpretability: Since deep learning models are frequently referred to as "black boxes," it might be challenging to understand how the model generates predictions. Future research could concentrate on creating more interpretable models that can justify their predictions, making the model's outputs simpler to comprehend and more reliable.

Deep learning models will need to adapt to recognize new kinds of false news as fake news detection techniques continue to develop. Future studies might concentrate on creating algorithms that can swiftly adjust to shifting fake news strategies and continue to work over time.

Evaluating real-world performance: While deep learning models' high accuracy in research settings is encouraging, it's crucial to assess how well they work in actual situations. Future studies could concentrate on applying these algorithms in real-world situations and assessing how well they work in identifying instances of fake news.

Overall, deep learning models have a lot of room for future research and development in the field of fake news detection, and these developments could have a big impact on the fight against the spread of false information and raising the standard of public information.

REFERENCES

[1] M.D.P. P Goonathilake, P.P.N.V Kumara,” CNN, RNN-LSTM Based Hybrid Approach to Detect State-of-the-Art Stance-Based Fake News on Social Media”,20th International Conference on Advances in ICT for Emerging Regions (ICTer),04-07 November 2020.

[2] Jamal Abdul Nasir, Osama Subhani Khan, IraklisVarlamis,” Fake news detection: A hybrid CNN-RNN based deep learning approach”,International Journal of Information Management DataInsights,April 2021, 100007.

[3] Rohit Kumar Kaliyar,”Fake News Detection Using A Deep Neural Network”, 2018 4th International Conference on Computing Communication and Automation (ICCCA),14-15 December 2018.

[4] Shalini Pandey, Sankeerthi Prabhakaran, N V Subba Reddy and Dinesh Acharya,”Fake News Detection from Online media using Machine learning Classifiers”,Journal of Physics: Conference Series, Volume 2161, 1st International Conference on Artificial Intelligence, Computational Electronics and Communication System (AICECS 2021) 28-30 October 2021, Manipal, India.

[5] Qing Liao, Heyan Chai, Hao Han, Xiang Zhang, Xuan Wang, Wen Xia, and Ye Ding,”An Integrated Multi-Task Model for Fake News Detection”,IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 34, NO. 11, NOVEMBER 2022.

[6] Sheng How Kong, Li Mei Tan,Keng Hoon Gan,Nur Hana Samsudin,”Fake News Detection using Deep Learning”, 2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE),18-19 April 2020.

[7] Chaitra K Hiramath, G. C Deshpande,”Fake News Detection Using Deep Learning Techniques”,2019 1st International Conference on Advances in Information Technology (ICAIT), 25-27 July 2019.

[8] Sachin Kumar,Rohan Asthana,Shashwat Upadhyay,Nidhi Upreti,Mohammad Akbar,”Fake news detection using deep learning models: A novel approach”,05 November 2019.

[9] Thota, Aswini; Tilak, Priyanka; Ahluwalia, Simrat; and Lohia, Nibrat (2018) "Fake News Detection: A Deep Learning Approach," SMU Data Science Review: Vol. 1: No. 3, Article 10.

[10] Rashki Hannah and Choi Eunsol et al. Truth of varying shades," Analyzing language in fake news and political fact-checking". In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2931–2937, Copenhagen, Denmark, 2017. Association for Computational Linguistics.

[11] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. Stop clickbait," Detecting and preventing clickbaits in online news media". pages 9–16, 08 2016.

[12] Alexandre Bovet & Hernán A. Makse," Influence of fake news in twitter during the 2016 US presidential election". Nature Communications, 10(7), 2019.

[13] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. "Recent trends in deep learning based natural language processing", 2017.

[14] Wang, W., Chen, Z., & Yang, X. (2020). Fake news detection with deep learning:

[15] Hussain, M., Ali, A., Javed, M. Y., & Baig, A. (2020). Fake news detection using deep learning techniques: A review. In 2020 International Conference on Computer, Control and Communication (IC4), (pp. 1-6). IEEE.

[16] Sun, H., Liu, Y., Guan, J., & Zhang, X. (2020). Fake news detection based on deep learning models. In 2020 IEEE International Conference on Information and Automation (ICIA), (pp. 470-475). IEEE.

[17] Yang, W., & Chen, J. (2020). A deep learning approach for fake news detection using ensemble methods. In 2020 IEEE 17th International Conference on Networking, Sensing and Control (ICNSC), (pp. 472-477). IEEE.

[18] Gao, S., Huang, Y., Hu, L., & Fang, Y. (2021). Fake news detection using deep learning models with feature selection.

[19] Zhao, R., Zhao, Z., Liu, X., & Zhang, H. (2019). Fake news detection based on deep learning methods. In 2019 International Conference on Robotics and Automation Sciences (ICRAS), (pp. 168-173). IEE

[20] Hu, J., Shan, Y., & Liu, Z. (2020). A multi-task deep learning model for fake news detection. In 2020 IEEE 12th International Conference on Communication Software and Networks (ICCSN), (pp. 347-352). IEEE.

[21] Zhang, Y., Qin, L., Zou, L., & Liu, Y. (2020). Fake news detection based on a long short-term memory network. In 2020 IEEE 7th International Conference on Cloud Computing and Intelligence Systems (CCIS), (pp. 564-569). IEEE.

[22] Shahriari, M., & Tabrizi, M. J. (2020). Fake news detection using deep learning and sentiment analysis. In 2020 15th International Conference on Computer Engineering and Systems (ICCES), (pp. 112-117). IEEE.

[23] Zhang, Y., Zhou, X., Wang, B., & Li, Y. (2021). A hybrid deep learning approach for fake news detection on social media.