

Fake Content Detection System for Multimodal Signals over Social Media

*A Thesis submitted in the fulfillment of the
requirements for the award of the degree of*

Doctor of Philosophy

submitted by

Sawinder Kaur

(Registration No. 901703008)

Under the supervision of

Dr. Parteek Kumar

Associate Professor, TIET

Dr. Ponnurangam Kumaraguru

Professor, IIIT-Delhi



Computer Science and Engineering Department

Thapar Institute of Engineering and Technology

Patiala-147004, India

September 2020

Contents

List of Figures	vi
List of Tables	ix
List of Abbreviations	x
Certificate	xiii
Acknowledgement	xiii
Abstract	xiii
1 Introduction	1
1.1 Fake content: An Overview	2
1.1.1 Types of Fake content	3
1.1.2 Fake content detection process	5
1.2 Need of fake content detection system	7
1.3 Applications of fake content detection system	10
1.4 Open challenges to detect fake content	12
1.5 Motivation	14
1.6 Research Gaps	15
1.7 Research Objectives	17
1.8 Research Methodology	17
1.9 Research contributions	21
1.10 Thesis Organization	24

2	Literature Review	29
2.1	Evolution of fake content detection analysis	29
2.2	Methodology	31
2.2.1	Review protocol	31
2.2.2	Research questions	31
2.2.3	Database sources	31
2.2.4	Inclusion-exclusion criteria	33
2.3	Extraction Outcomes	35
2.4	Datasets	36
2.4.1	Textual dataset	36
2.4.2	Non-textual dataset	37
2.5	Open-source tools	40
2.6	Classification of fake content detection techniques	43
2.6.1	Social spam detection	44
2.6.2	Review content-based models	46
2.6.3	Hidden representations	58
2.6.4	Location-based models	60
2.7	FCD analysis for textual data	60
2.7.1	Research studies related to fake textual content	61
2.7.2	Research studies related to clickbait headlines	64
2.8	FCD analysis for non-textual data	66
2.8.1	Research studies related to unreliable image content	67
2.8.2	Research studies related to deepfake clips	76
2.9	Discussions	104
3	Implementation of Fake News Article Detection System	106
3.1	Contributions	107

3.2	Problem Statement	108
3.3	Methodology	109
3.3.1	Data Collection	110
3.3.2	Pre-processing	112
3.3.3	Feature Generation	114
3.4	Classification Algorithms	121
3.4.1	Naïve Bayes (NB)	123
3.4.2	Linear Model	124
3.4.3	Support Vector Machine (SVM)	124
3.4.4	Neural Network (NN)	126
3.4.5	Decision Tree (DT)	126
3.4.6	Ensemble methods	127
3.5	Evaluation Phase	127
3.5.1	Major findings	136
3.6	Proposed multi-level voting model	137
3.7	Evaluating the performance of the proposed model	140
3.8	Comparison with Existing Works	142
3.9	Discussions	146

4 Clickbait Identification using Two-phase Hybrid CNN-LSTM

	Biterm Model	148
4.1	Contributions	149
4.2	Problem statement	150
4.3	Data collection	151
4.4	Architecture of the proposed system	154
4.4.1	Pre-processing	156
4.4.2	Embedding layer	158

4.4.3	Deep-learning based hybrid classifier	160
4.4.4	Biterm Topic Model (BTM)	164
4.5	Experimental results	166
4.5.1	OCR performance	167
4.5.2	Classification performance	169
4.5.3	Cluster analysis to identify the types of clickbait head- lines	171
4.6	Comparison with existing systems	175
4.7	Discussions	180

5 DeepFakes: Temporal Sequential Analysis to Detect Face-Swapped Video Clips using C-LSTM 183

5.1	Motivation	186
5.2	Contributions	187
5.3	Problem statement	188
5.4	Deepfake videos exposed	189
5.4.1	Training and generation of video using DeepFaceLab	189
5.5	Deepfake tools	191
5.6	Data Collection	191
5.6.1	Experimental Setup	191
5.7	Methodology	197
5.7.1	Image pre-processing	201
5.7.2	Convolutional-Long Short Term Memory (C-LSTM)	202
5.8	Evaluating C-LSTM model	205
5.9	Major findings	209
5.10	Discussions	210

6	A Cross-Platform Application to avoid Fake Web Trap	212
6.1	Contributions	212
6.2	Methodology	214
6.3	Preliminaries	217
6.3.1	Features of the proposed system	217
6.4	System architecture	219
6.4.1	QuoteExaminer	221
6.4.2	Fake news article	224
6.4.3	Clickbait Detector	226
6.4.4	Deepfake swap	228
6.5	Implementation	231
6.6	Discussions	232
7	Conclusion and Future scope	234
7.1	Conclusions	234
7.2	Future Work	237
	List of Publications	238
	References	239

List of Tables

2.1	Formulation of research question for literature review	32
2.2	Advance keyword-based search for extracting research articles	34
2.3	Count of publications according to citations	35
2.4	Statistics of existing textual fake content datasets	37
2.5	Statistics of existing non-textual fake content datasets	38
2.6	Statistics of existing deepfake datasets	39
2.7	Summarized review of research studies related to spam detec- tion techniques	47
2.8	Summarized review of research studies related to lexical model techniques	49
2.9	Summarized review of research studies related to syntactic model techniques	51
2.10	Summarized review of research studies related to semantic techniques	53
2.11	Summarized review of research studies related to content and style similarity techniques	56
2.12	Summarized review of research studies related to hidden rep- resentation techniques	59

2.13 Summarized review of research studies related to location-based techniques	61
2.14 Comparative analysis of research studies related to fake textual data	65
2.15 Comparative analysis of research studies related to fake images seen over social media	74
2.16 Comparative analysis of research studies related to deepfake detection	79
2.17 Statistics of existing dataset to detect deepfake using machine learning-based approach	82
2.18 Comparative analysis of existing approaches to detect deepfake using machine learning based approach	83
2.19 Statistics of existing dataset to detect deepfake using deep learning based approach	86
2.20 Comparative analysis of existing approaches to detect deepfake using deep learning-based approach	94
3.1 Statistics of collected corpus	111
3.2 Mean distribution of labeled articles for News Trends, Kaggle and Reuters corpus	114
3.3 Sparse matrix representation using a Count-Vectorizer feature extraction technique	116
3.4 A confusion matrix representation	128
3.5 Comparative analysis of accuracy measure using machine learning classifiers	129
3.6 Comparative analysis of precision metric using Machine Learning classifiers	131

3.7	Comparative analysis of recall metric using Machine Learning classifiers	132
3.8	Comparative analysis of specificity metric using Machine Learning classifiers	133
3.9	Comparative analysis of F1-score metric using Machine Learning classifiers	133
3.10	Comparative analysis of the proposed multi-level voting model with voting classifier	141
3.11	Comparison of existing models with a proposed system for Reuters corpus	143
3.12	Comparison of existing models with a proposed system for Kaggle corpus	144
3.13	Comparison of existing models with a proposed system for News Trends corpus	145
4.1	Statistics of collected datasets	152
4.2	Mean distribution of collected datasets	154
4.3	Embedding layer specifications of the proposed model	158
4.4	Number of parameters in the hybrid CNN-LSTM model using <i>GloVe</i> embeddings	164
4.5	Results obtained after performing OCR and pre-processing on Dataset 3	167
4.6	Comparative analysis of the hybrid CNN-LSTM architecture with and without pre-trained vectors using performance metrics	171
4.7	Coherence score of top 5, 10 and 20 words for Dataset 1, Dataset 2 and Dataset 3 among nine clusters retrieved using BTM to analyse the quality of clusters	172

4.8	Description of identified topics	173
4.9	Distribution of headlines	174
4.10	Comparison of the experimental results of our proposed model with the existing systems	176
5.1	Summary of existing deepfake tools	193
5.2	Summary of configurations required for the collection of dataset	194
5.3	A sample of 20 Person of Interest (POI) chosen for source and destination video clips from our collected dataset with high- quality downloads	195
5.4	A sample of 20 Person of Interest (POI) video clips from our collected dataset with total time taken to train the model, total loss after training the model, total number of frames retrieved from these clips.	196
5.5	Comparative analysis of performance measures using MesoNet, Capsule, CNN and the proposed deep temporal-based C-LSTM architecture on our collected dataset	207
5.6	Comparative analysis of performance measures using Celeb- DF, DFDC and ground dataset using proposed deep temporal- based C-LSTM architecture	209
6.1	Analysis of the proposed systems	218

List of Abbreviations

Tf-Idf	Term Frequency- Inverse Document Frequency
HV	Hashing-Vectorizer
CV	Count-Vectorizer
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
LSTM	Long Short Term Memory
NLP	Natural Language Processing
BTM	Biterm Topic Model
OSNs	Online Social Networks
C-LSTM	Convolutional-Long Short Term Memory
PWA	Progressive Web Application
URL	Uniform Resource Locator
ML	Machine Learning
DL	Deep Learning
OSM	Online Social Media
SVM	Support Vector Machine
BoW	Bag of words
K-NN	K-Nearest Neighbor
API	Application Programming Interface
Tf	Term frequency
LSVM	Linear Support Vector Machine
DT	Decision Tree
SGD	Stochastic Gradient Descent
LR	Logistic Regression

LDA	Latent Dirichlet Allocation
MCL	Markov Clustering
POS	Part of Speech
LIWC	Linguistic Inquiry Word Count
MAP	Mean Average Precision
VSM	Vector Space Modelling
RST	Rhetorical Structure Theory
CSI	Capture Score Integrate
LSBN	Location Based Social Net
FCD	Fake Content Detection
OCR	Optical Character Recognition
DIA	Document Image Analysis
CFFN	Common Fake Features Network
JSON	JavaScript Object Notation
HTTP	Hypertext Transfer Protocol
GAN	Generative Adversarial Network
DFDC	Deepfake Detection Challenge
DFD	Deepfake Detection
FF-DF	FaceForensics Deepfake
WOT	Web of Trust
ALPR	Automatic License Plate Recognition
LOG	Laplace Gaussian Operator
CRM	Copy Rotate Move
CSR	Class Sequential Rules
LSGAN	Least Square Generative Adversarial Network
DCGAN	Deep Convolutional Generative Adversarial Network

PGGAN	Progressive Growing Generative Adversarial Network
AI	Artificial Network
MLP	Multi Layer Perceptron
LRCN	Long-term Recurrent Convolutional Network
GDB	Gradient Descent Boosting
PLSA	Probabilistic Latent Semantic Analysis

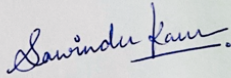
Acknowledgement

First of all, I express my gratitude to the Almighty, who blessed me with the zeal and enthusiasm to complete this work successfully. I would like to thank my supervisor Dr. Parteek Kumar and Dr. Ponnurangam Kumaraguru for their valuable guidance. They have always been there to guide me towards the right direction whenever I got stuck with the ideas. I want to thank them for their valuable evenings and week-ends that they spent with me in finalizing the writings of this thesis work. I will always remain indebted to them for showing me the path. I also want to thank Mrs. Kumar to provide a conducive environment in her home during discussions on this proposed work. I express my gratitude to the Doctoral Committee for monitoring the progress and providing valuable suggestions for improvement of my research work. I feel very proud and thankful to the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala for providing all the resources for a good research work. I am also very thankful to Dr. Maninder Singh (Head CSE department) and all my colleagues for their constant support during my research work. This thesis would have been impossible without the support of my family. My deep regards to my father Mr. Arvinder Singh Vohra, my mother Mrs. Ramanjit Kaur and my sister Ms. Vibhleen kaur for their patience and love. Without them this work would never have come into existence. They have provided me with lessons on honesty and ethics and their humbleness and patience have always amazed me. At the end I wanted to thanks all my friends at Sachendra Singh, Sandeep Saharan, Sujata Rani, Sonal Kukreja and Priya who have been stress busters for me.

(Sawinder Kaur)

Certificate

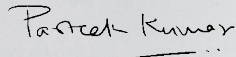
I hereby certify that the work which is being submitted in this thesis entitled "Fake content detection system for multimodal signals over social media", in fulfilment of the requirements for the award of the degree of DOCTOR OF PHILOSOPHY submitted in Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Parteek Kumar, Dr. Ponnurangam Kumaraguru, refers work of other researchers which are duly listed in the reference section. The matter presented in this thesis has not been submitted for the award of degree in any other University.



(Sawinder Kaur)

Regd. No. 901703008

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



(Parteek Kumar)

Associate Professor, Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala-147004 (INDIA)



(Ponnurangam Kumaraguru)

Professor, Computer Science and Engineering Department
IIIT-Delhi, Delhi- (INDIA)

Abstract

Fake contents can be easily created and spread through the social media and web-based platforms, resulting into the widespread of real-world impact. Novel digital technologies make it increasingly difficult to distinguish between real and fake media. One of the most recent developments contributing to the problem is the emergence of fake contents in form of images, videos, articles, posts, clickbaits which are hyper-realistic to depict the things that never happened. Coupled with the reach and speed of social media, such fake contents can quickly reach millions of people and have negative impacts on the society. To develop fake content detection tools, characterizing of how fake data proliferates over social platforms and why it succeeds in deceiving readers are critical to develop such efficient tools for early detection.

The research work presents the description about the general process of fake content detection using various multimodal signals (articles, URLs, posts, images, videos) over social media platforms. A literature review related to the field of fake content detection for textual and non-textual content is depicted in this research work. The current status of fake contents over social media in form of multimodal signals is classified in two categories (textual and non-textual). The periodical evolution seen in the field of fake content generation and research studies on the basis of publications has been analysed. Further, review protocol is followed and presented, selected sources of publications, retrieved research papers on the basis of inclusion-exclusion criteria. This research approach will help to make the findings available in a systematic way for assisting the researchers working in similar area to select the most appropriate techniques to identify fake content for textual and non-textual datasets.

This thesis presents the multi-level architecture of fake news article detection system using three feature extraction techniques for efficiently identifying the category of news articles using machine and deep learning based techniques. To train the framed architecture, various benchmark datasets of fake and real articles have been collected from Reuters, News Trends and Kaggle websites. Different machine learning based models and neural networks trained on the collected datasets have been compared with the proposed model on the basis of accuracy, recall, precision, F1-score, AUC-ROC performance metrics. The proposed multi-level voting model not only helps to improve the accuracy but also helps to increase the efficiency of the model. The experimental analysis shows that the proposed model outperforms the traditional machine learning algorithms for fake news article classification. Further, to process with both textual (clickbait headlines) and non-textual (headlines seen over posts in form of images) dataset, a novel two-phase hybrid Convolutional neural network- Long Short Term Memory Biterm approach has been proposed for modelling short topic content. The hybrid model when implemented with pre-trained GloVe embedding yields the best results based on accuracy, recall, precision, and F1-score performance metrics. Eight types of clickbait such as Reasoning, Number, Reaction, Revealing, Shocking/Unbelievable, Hypothesis/Guess, Questionable, Forward referencing are classified in this work using the Biterm Topic Model. Also, a ground dataset of non-textual (image-based) data using multiple social media platforms has been created using human annotations. The textual information has been retrieved from the images with the help of pre-processed OCR tool. A comparative study is performed to show the effectiveness of the proposed model.

Deepfake enthusiasts have been using neural networks to produce convincing face swaps. The impact of deepfakes has been alarming with politicians,

senior corporates and world leaders being targeted by nefarious actors. To detect face-swapped deepfake video clips, a novel approach using the temporal sequential frames is proposed. The proposed approach uses the forged video to extract the frames at first level followed by which a deep depth-based Convolutional Long Short Term Memory model to identify the fake frames at second level. Also, the proposed model is evaluated on newly created ground truth dataset of forged videos using source and destination video frames of famous politicians. The newly ground dataset is created with the help of DeepFaceLab tool using source and destination videos from YouTube website. Experimental results demonstrate the effectiveness of the proposed model.

This thesis also presents the developed Progressive Web Application of the proposed systems. Various features of the proposed schemes has been discussed that are embedded in the system. This web application promotes the exchange of information among its users. It serves the purpose to expedite the users to detect different types of fake or unverified content seen in the form of quotes, videos, articles, clickbaits over social media platforms. The goal of developing such an application is to outreach the social media users to give the power to analyse the reliable, trustworthy and real content for social development. Inappropriate use of unreliable words can also be detected through the developed application. This system can also help research teams, end-users, general public in analysing the fake contents over social media platforms.

Chapter 1

Introduction

A growing interest related to fake content detection has attracted many researchers as fake information is circulated through online social media platforms [1] such as Facebook, Twitter, *etc.* The fake content is spreading at a faster pace to gain popularity over social media [2], to distract people from the current critical issues. Most of the people believe that the information they receive from various social media sites is reliable and true, *i.e.*, people are inherently truth-biased. Also, people easily trust and want to believe in what they actually interpret in their minds, *i.e.*, confirmation-biased. In general, it has been analysed that humans are unable to recognize deception effectively.

This chapter provides the overview of the thesis work. It discusses the various types of fake content spread over social media platforms, the generic flow to detect the fake contents seen in form of news articles [3], clickbait headlines [4, 5], misquoted images, deepfake video clips. The chapter also focuses on the need for automatic detection of such content disseminated over Internet. The real-time applications of such fake content detection sys-

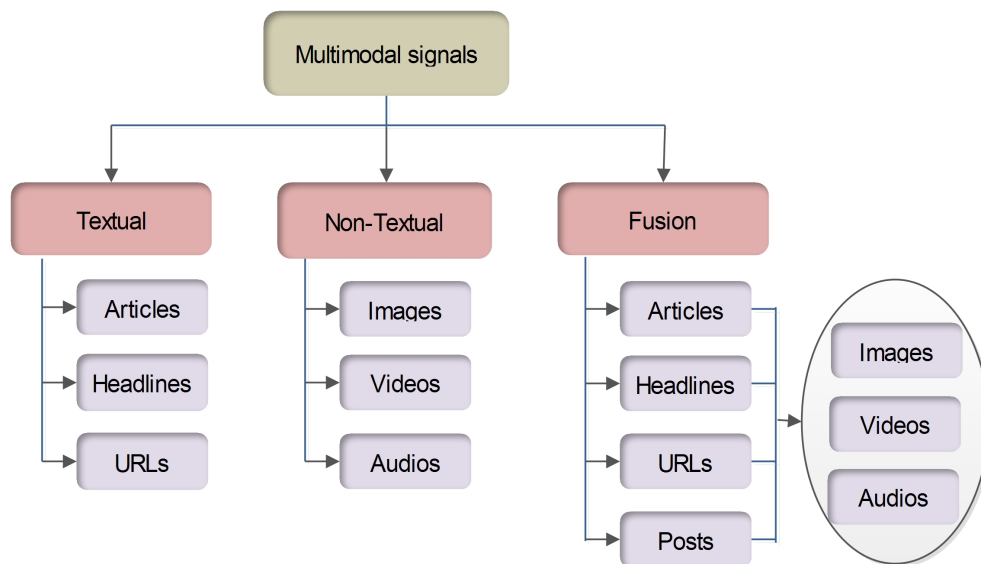


Figure (1.1) Different types of content spread on social media platforms

tem, major challenges, motivation of the research to propose the problem statement to build fake content detection system for multimodal signals over social media are also discussed in this chapter. It concludes by discussing the structure of the thesis along with its objectives, methodology and contributions.

1.1 Fake content: An Overview

Fake content seen in various form of news, is an inaccurate made up story with an intention to mislead, gain attention, deceive or damage a reputation. The type of content shared or spread over social media websites such as images, videos, articles, links, posts, blogs, *etc.*, are known as multimodal signals as shown in Figure 1.1.

Traditionally media outlets, journalists required to follow strict codes of

practise to get news from trusted sources. Nowadays, the smart genesis has deviated from newspaper, books to the online social platforms for gathering knowledge. Thus, making the Internet to enable a new way to consume, share and publish the low editorial level contents. Many people rely on social media sites and networks, thus misleading themselves to understand the authenticity of published contents in form of quotes, articles, images, posts, videos, *etc.* Such overloaded content on social media platforms cause confusion, influence user's views, push political agendas and can lead to a profitable business for online publishers. Fake information can deceive people by making their appearance similar to the trusted web addresses using similar website names. Dissemination of such fake contents changes the user's interests and beliefs by deflecting the attention of users from real to fake news.

1.1.1 Types of Fake content

Fake news is everywhere, but no one seems to agree on what it actually means. Distinct categories of fake news are discussed below.

- i) **Deliberate Misinformation:** It is misleading information or intentionally fake which is spread in a calculated way to deceive targeted users. A person, who lived in Canada has been falsely accused across social media. In November 2016, the person was misinterpreted as being one of the bombers for attacks in Paris through altered image to play off the fears and emotions of people around the world.
- ii) **Clickbait Headlines:** To grab the reader's attention with a purpose to make them click on the fake contents are known as clickbait headlines on the Internet [6, 7]. Users who set up fake sites generate huge revenues and clicking on such websites results in bombarded ads. Clickbait headlines



Figure (1.2) Eye catchy headlines

create an enthusiasm among users to get trapped to click the headlines as shown in Figure 1.2.

- iii) **False Headlines:** To draw a reader in, false headlines are intentionally exaggerated. In such headlines title of the article may not match the context of stories, the headline can be read as one way and state something different as a fact [3]. This type of fake content is untrue at worst and misleading at best. Famous personalities tweeted a Times of India story with a distorted headline, before deleting the image, it got spread and was already used multiple times on social media.
- iv) **Satire:** Satirical fake content sources use ridicule, irony, humor, false information and exaggeration to comment on current events [8]. The satirical content is purposely twisted to comment on the state and issue of the society.
- v) **False context:** It is seen when genuine content is shared with false context [6]. A Facebook post was showing a picture of a man and girl smiling with their faces smeared with blood went viral with a false content creating anger among society.



Figure (1.3) Misleading content seen on Facebook

- vi) **Manipulated Content:** When imagery or genuine information is manipulated to deceive. In Figure 1.3, a manipulated content is seen where “GUTOM LANG ANG INABOT KO # NEVER AGAIN” is shared instead of “END MARTIAL LAW # NEVER AGAIN”. When this page was shared on Facebook, MemeBuster website added that it was shared thousands of times after the rally against dictatorship.
- vii) **Deepfake content:** It is a type of video or image forgery developed to spread misinformation, invade privacy, and mask the truth using advanced technologies such as trained algorithms, deep learning applications, and artificial intelligence.

The generic process to detect such content is discussed in the next section.

1.1.2 Fake content detection process

A generic approach to perform fake content detection includes five phases as shown in Figure 1.4.

- i) **Data collection:** Huge amount of content in form of unstructured data (images, posts, articles, clickbaits, videos, audios) is available on social



Figure (1.4) Generic process to detect fake contents

media platforms. There are many open source web-scraping tools available online to gather the data from OSM platforms. Every day enormous data is generated by the social media users (in form of shares, likes, retweets, comments, views, opinions). Various social media APIs are available to extract the unstructured data, further analysis can be performed on the extracted data.

- ii) **Pre-processing:** In the pre-processing phase, two types of dataset are dealt, one as textual and the other as non-textual. Non-semantic words such as prepositions, conjunctions, pronouns, *etc.*, also known as stop words are removed from the textual document as they provide very little or no information about fake content. The redundant data in form of textual strings is removed from the document using a regular expression. The *regex* and *pandas* library can be used to perform the pre-processing task. For, non-textual data (images and videos), *ImageDataGenerator* class of *Keras* library can be used to extract the frames from video clips, image rescaling, shear mapping, zooming augmentation and horizontal flip can be further performed to remove the noise from the collected data. The purpose of pre-processing is to convert noisy data from high dimensional features to the low dimensional space to achieve better performance.
- iii) **Feature extraction:** The whole process of converting the textual document into numerical feature vector is called as vectorization. In order to extract numerical features from a textual document, tokenization,

counting and normalization is done. During tokenization, each word is given a unique integer ID, following which occurrence of tokens is counted and then normalization of such tokens takes place. To deal with both textual and non-textual data, various word embeddings like Gensim (NLP library), which provides an amazing wrapper to adopt different pre-trained word embedding models which include *GloVe* and *Word2Vec*. Such pre-trained embedding models can be used to provide a better feature representation of words.

- iv) **Fake content detection:** This phase deals with the classification of both textual and non-textual data. In this phase, various machine and deep learning algorithms like Random Forest, SVMs, Ensemble models, Linear models, Naïve Bayes, Decision Tree, Support Vector Machines, CNN, C-LSTM, RNN can be used. Further, fake file extraction analysis can be performed.
- v) **Output analysis:** In this phase, some meaningful information can be extracted from the textual and non-textual data, predicted as fake or real. Various interesting characteristics can be analysed. The results of this phase can also be added to system which may help end-user to analyse the categorization of the predicted news content.

There is an urge to build such real-time fake content detection system, which is discussed in the next section.

1.2 Need of fake content detection system

The extent of its impact reaches far beyond the political realm, with its influence ranging from distorting financial markets to facilitating political

extremism. Some of the impacts seen during the spread of fake contents are listed below.

- i) **Political Influence:** During 2016 US Presidential Election campaign, spread of fake news was accelerated dramatically. Groups of teens in Macedonia were found to have been intentionally creating pro-Trump fiction, profiting from their shareability [9]. A Russian propaganda group was found to have been creating pro-Trump fake news in order to give the impression that there was a strong grass-roots movement legitimately supporting his campaign. The influence of these activities is hard to quantify, but the emergence of fake news creation is seen as a way to shape the political landscape is an undeniably dangerous issue [10, 6].
- ii) **Encourages Mistrust in Legitimate Media Outlets:** Politicians have been able to capitalize on the widespread awareness of fake news to discredit media outlets that are critical of their work [9]. In both UK and US, political parties have used the term fake news to diminish the authenticity of reporting on either their own policy plans or that of their political opponents. The spread of fake news has also diminished the value of expert knowledge.
- iii) **Influences the Financial Markets:** In 2015, fake reports of peace between France and England sent share prices in the London Stock Exchange soaring by 5% almost immediately. Most of this news was spread using platforms that include Twitter [11, 12, 13] and Telegram as a way to instil confidence in particular cryptocurrency and therefore, artificially inflate their value to the benefit of currency holders.
- iv) **Bad for Business:** Big corporations are becoming the targets of politically motivated fake news campaigns increasingly often.

- v) **Damaging to Personal Reputation:** The celebrity death hoax is an example of fake news proliferation that everyone is familiar with, but it's not the only type of false stories that are written about the stars. Many different celebrities were targeted by fake news campaigns designed to discredit them. For example, Amitabh Bachan was declared dead when went for abdomen operation in hospital.
 - vi) **Pornography:** Many deepfake clips [14] on the Internet are featured with fake porn clips of female celebrities for earning money through social media platforms. The first deepfakes featuring Indian celebrities was published by pornhub user 'blackandwhitepanda'. The videos come with a deepfake disclaimer, but are hugely disturbing to society as it consists of non-consensual content.
 - vii) **Defaming of celebrities:** Amateur deepfake technology has grown to create a virtual actor for the films. The human eyes cannot detect the difference between real and virtual videos. This creates difficulties for the famous celebrities as their fake dubbed videos, audio clips passing shocking/irrelevant statements are disseminated over social media platforms. Thus, making them responsible to answer to media, journalists, *etc.*
 - viii) **Damage to Society:** An elderly Chinese woman was falsely accused in 2017 for being a part of a child kidnaps team. After verification and investigation, a woman was identified as a regular customer at a shopping mall. Later the woman was proved innocent but was of no use as the damage was done to her. The elderly woman shied away from people and was afraid of leaving her home.
- Fake content is shared on various social media platforms [15, 11, 12, 13].

Different applications of fake content detection system are discussed in next section.

1.3 Applications of fake content detection system

There are numerous applications of fake news detection analysis. Some of the most important applications are discussed below.

- i) **Control of fake spread during elections:** Fake news was very prevalent during the 2019 Indian general elections. Misinformation was prevalent at all levels of society during the build-up to the election. The elections were called by some as “India’s first WhatsApp elections”, with WhatsApp [15] being used by many as a tool of propaganda as “misinformation was weaponized”. The accounts shared such unauthorized content on Facebook, Twitter [11, 12, 13], which was analysed through fake news detection tools were deleted.
- ii) **Corona pandemic:** Misinformation related to Covid-19 is in the form of social media messages related to home remedies that have not been verified, fake advisories and conspiracy theories. Two people got arrested for spreading fake news about the coronavirus pandemic. An appeal was made by Indian Prime Minister to not believe any rumours related to the Covid-19 pandemic. Many scientists are working together to debunk such fake information by building and using fake tools and involving human help.
- iii) **Terrorism:** Due to the broad reach of social media platforms such as

YouTube, Facebook and Twitter [11, 12, 13], terrorist groups and individuals have increasingly used social media to further spread their messages. Attempts have been made by various governments and agencies to thwart the use of social media by terrorist organizations. Terror groups take to social media because it's cheap, accessible, and facilitates quick access to a lot of people. After spending some time, and following their networks through social media platforms using crowdsourcing, experts can analyse their hypocritical, violent ideas.

- iv) **Natural calamities:** During natural disasters or crises, users on social media tend to easily believe contents of postings related to the events, and retweet the postings with a hope of reaching to many other users. Unfortunately, there are malicious users who understand the tendency and post misinformation such as spam and fake messages with expectation of wider propagation [16]. Fake images and spam messages are widely spread during natural disasters on social media [17].
- v) **Fighting riots:** Fake accounts are a preferred means for malicious users of online social networks to send spam, commit fraud, or otherwise abuse the system [17]. A single malicious actor may create dozens to thousands of fake accounts in order to scale their operation to reach the maximum number of legitimate members. Detecting and taking action on these accounts as quickly as possible is imperative in order to protect legitimate members and maintain the trustworthiness of the network. The government and police always remain very vigilant against any communal clashes, since it can lead harm to a society in terms of human loss, faith, economy and pride of the nation. So, the fake accounts are a preferred means for malicious users of online social networks to send spam,

commit fraud, or otherwise abuse the system.

Automatic fake content detection system can be used to control the spread of fake information seen in form of images, articles, blogs, videos, audios over various social media platforms [15, 11, 12, 13]. Various challenges faced to detect fake content are discussed in next section.

1.4 Open challenges to detect fake content

Many different challenges were faced while detecting fake news content seen in the form of multimodal signals over social media platforms. Some of these challenges are discussed below.

- i) **Traditional algorithms:** Several algorithms have been developed for detection in different domains. However, they are not directly comparable to each other due to the lack of large-scale publicly available datasets of false information, spanning fake reviews, hoaxes, and social media rumors. This prevents a benchmark comparison between different categories of algorithms.
- ii) **Unstructured data:** A huge amount of data is seen over social media platforms and over 70 % of the information accounts unstructured data (images embedded with textual information, audios, videos). Such platforms contain information from sources like newspapers, journals, authenticated or unauthenticated news websites, *etc.* As the information on Internet is seen in different formats, such type of un-structured data makes the analysis more complex.
- iii) **Unavialable datasets:** Deepfake benchmark datasets related to famous politicians are not available, which requires lots of time to invest for

collecting the original video speech clips. No standard splitting method has been defined prior to follow a generic approach for dataset split of videos at frame-level or pixel-level. Many constraints are required to be considered during collection of source and destination video clips to generate deepfake face-swap clips.

- iv) **Mode of education:** Reducing the damage of false information is an essential direction that is open for research. Recent research studies show that educating people against possible manipulation strategies used in false information is effective in improving human detection skills. Further research in finding effective educational strategies to vaccinate people against believing false information, and how to scale these strategies to millions of users that use social platforms is necessary.
- v) **Third generation deepfake dataset:** Implementing the models on third generation deepfake dataset is quite challenging. Third generation datasets not only have more than an order of magnitude larger number of frames and videos than the second generation, but has with better quality including the agreement from individuals involved appearing in the collected dataset.
- vi) **Match with false content:** False information can be checked by matching it against knowledge base of complete information. This direction poses numerous challenges like successfully creating and maintaining this knowledge-base, and ascertaining data quality.
- vii) **Appropriate feature extraction:** Natural language understanding and information extraction techniques must be developed to automatically extract information. Finally, we would require capable informa-

tion matching algorithms in order to check if the extracted information matched with the existing information in our collected knowledge-base.

1.5 Motivation

With the growth of online and social media forums, the spread of fake news is causing an easy blend with real or authenticated information. It leads to distort the awareness, decision making, gain attention, deceive or damage a reputation by influencing perception of society. Due to which posting links, posts in form of images, videos, audios over social media have proven to be an easy task by creators of such manipulated content. Distributors and creators of fake content have political and social agendas behind its spread.

Nowadays, an increased popularity is seen for video and audios manipulation on social media platforms, which are known as deepfakes. The deepfakes uses advanced techniques such as machine learning and deep learning to manipulate audio and general content with a high degree of symbolic manipulations. Deepfake videos have been used to create pornographic videos of celebrities (known as revenge porn), fake news, malicious hoaxes, fake surveillance, political distress, blackmailing of an individual, and fake terrorism and financial fraud which create political tensions in daily-life among government entities.

It was observed that during US president election [9], most of the widely spread articles on social media were fake. Recently a fake video related to Kerela battling with floods was viral on social media platform (Facebook). Also during India's national election (2019), various WhatsApp groups (> 900,000) were created to disseminate the fake information regarding India's ruling party [10]. Other examples related to videos clips, where faces of

politicians are swapped when they deliver speech during election campaign cannot be examined by the general public to analyse the video clip as deep-fake or real. It is proposed that with the upcoming 2020 US elections will be influenced by fake contents (audios, videos, images) spread over social media through shares, retweets.

Several states like New York, Virginia, California, Texas have introduced different legal actions to control the problems related to fake news. After introducing various deepfake [18] prohibition acts by many countries, still no restriction is seen for deepfake creation. The spread of such content creates a huge massive impact on society and individuals leading to an imbalance of the news ecosystem. Deepfake videos are spreading at a faster pace and human-eye cannot recognize the difference between fake and real multimodal signals. So, it is important to build an automated real-time fake content detection system. Where a single system will be proposed to provide a novel solution to detect fake contents in form of articles, clickbaits [6], videos and images over social media platforms to reduce the risks posed by the spread of such contents.

1.6 Research Gaps

In context to the literature review conducted on the existing research done on fake news detection which is relatively a recent phenomenon regarding the interest being raised by society, following gaps have been observed, as listed below:

- i) Existing fake content detection models are built using supervised machine learning algorithms where manual hand-crafted features [7] are to be extracted which is very time consuming and inefficient method to

achieve the best accuracy.

- ii) No benchmark dataset is available for the non-textual clickbait headlines, political deepfake video clips related to politicians speech [19].
- iii) No pre-processing approach is embedded with the existing text extraction tools to extract accurate textual data from the images [20], which may increase the overall performance of the models.
- iv) It is important to identify the type of false headlines, whereas no such existing system is build to analyse the type of short textual headlines spread over social media platforms.
- v) The existing models developed do not score the validity of the source, which makes it difficult to detect real or fake content during real-time [21].
- vi) No such model is built for the verification of domain name [22].
- vii) The existing algorithms do not put emphasis for detecting the degree to which content is invalid.
- viii) The existing systems to detect fake contents do not give accurate results for real-time data collection.
- ix) The existing real-time systems to detect fake contents are relied on images, posts, but no such system exists which can detect fake video clips [7].
- x) There is no existing model available for the prediction of the deepfake video clips on the basis of temporal sequences and inconsistencies be-

tween the retrieved frames to detect the fake face-swapped video clips of politicians within an initial time-frame.

- xi) Individual systems are available to check the fact-based news, no such embedded system is available to detect the unverified quotes seen in form of images, fake news articles using content (body) or URLs to extract the textual data, clickbait headlines, face-swap deepfake video clips within a single application.

The research gaps studies show new avenues for future work. Existing techniques studied so far provides a direction to be followed further for quantitative and qualitative research.

1.7 Research Objectives

The following objectives of the proposed work are:

- i) To study and analyse the existing fake content detection techniques.
- ii) To perform data acquisition for collecting dataset from different Online Social Networks (OSNs).
- iii) To propose and implement fake content detection system for multimodal signals.
- iv) To test and validate proposed fake content detection system.

1.8 Research Methodology

A proper research methodology has been followed to achieve the objectives of the research work.

i) **To achieve the first objective**

- A detailed analysis of different types of fake contents spread over social media has been performed to get the knowledge of existing fake content detection systems.
- A review study related to the research work has been followed from the various renowned electronic conferences (national and international) and databases related to the research field.
- An advance keyword based search has been followed to extract the relevant research articles from the electronic-databases.
- To perform a survey of research studies, set of research questions has been formulated.
- The impact of research studies for different multimodal signals (quotes images, clickbait headlines, fake news articles, deepfake video clips) over social media platforms have been identified.
- Various open source tools and resources available have been identified for detection of fake news contents.

ii) **To achieve the second objective**

- The existing standard dataset related to news articles from the authenticated websites (News Trends [23], Kaggle [24] and Reuters [25]) have been collected to evaluate the performance of the proposed model to detect fake news articles efficiently.
- A ground dataset has been prepared from the Facebook page and Reddit website to extract clickbait headlines from collected images with the help of human annotations. Also, two standard datasets [26, 27] were collected to perform the experiment.

- Top 30 words are extracted to analyse the distribution of fake and real terms for both clickbait headlines and fake news article content.
- Statistical analysis of collected datasets for news articles and clickbait headlines with negative and positive instances has been performed.
- The statistical test known as Z-test is used to compare the mean distribution of two samples (clickbait and non-clickbait; fake news articles and real news articles).
- A ground truth dataset (deepfake and real frames) has been prepared for deepfake video clips. Data is generated by swapping famous politicians face to the pre-existing video clips for detection of deepfake face-swapped video clips.

iii) **To achieve the third objective**

- The system has been trained on various machine learning and deep learning approaches using different feature extraction techniques to identify fake news articles, clickbait headlines, deepfake video clips in Python.
- Proposed an ideal multi-level voting classifier to verify the effectiveness of the ensemble model to classify fake news articles.
- A system has been proposed to detect clickbait headlines is proposed. It works under a two-phase structure. In the first phase, the headlines of textual and non-textual data are fed to the CNN-LSTM model and in the second phase, the classified clickbaits are fed to the Biterm Topic Model (BTM). It uses a biterm co-occurrence of words to cluster similar clickbait headlines to identify the types of

clickbait.

- A deep temporal-based C-LSTM model has been proposed to identify deepfake face-swapped video clips. In the proposed approach, the forged frames from deepfake videos are extracted using OpenCL and a frame-level pre-processing is performed at the first level. The pre-processed frames are further fed to a C-LSTM algorithm to analyse the deepfake (face-swapped) videos.
- Prediction of the deepfake video clips on the basis of the temporal sequences and inconsistencies between the retrieved frames from the LSTM layer of C-LSTM model to build a highly efficient model.
- A progressive web application is developed to expedite the users to detect different types of fake or unverified content seen in the form of quotes, deepfake videos, news articles, clickbaits over social media platforms.
- The system is able to extract the real author name from the misquoted images; extract textual content from the URL to extract necessary features to identify fake news articles; extraction of textual content from the URL (clickbait links) to identify a clickbait headlines and graphical analysis can be performed by the system to classify the type of clickbait headlines; a user-friendly analysis has been shown by the system on the retrieved frames on the basis of time interval to identify deepfake face-swap video clips.

iv) **To achieve the fourth objective**

- The efficiency of the proposed systems has been compared with the various existing state-of-the-art models on the basis of various standard performance metrics.

- The conventional performance measure has been evaluated using confusion matrix cells for all three proposed systems.
- Word clouds have been generated to retrieve top words of real and fake textual content to compare with the existing techniques.
- Various graph analysis for training and testing accuracy and loss for deep learning models have been evaluated and compared with other state-of-the-art models.

1.9 Research contributions

Our main contribution of this thesis is the development of Progressive Web Application (PWA) to detect fake content over social media platforms. The overall contribution of this thesis is as follows.

- i) An extensive literature review on fake news articles, clickbait headlines, both deepfake video clips generation and detection have been explored highlighting the existing tools, key challenges, and comparison with state-of-the-art models in our thesis work.
- ii) The existing dataset related to news articles from the authenticated websites (News Trends, Kaggle and Reuters) have been collected. A ground dataset has been prepared from the Facebook page and Reddit website to extract clickbait headlines with the help of human annotations.
- iii) A ground truth dataset related to real and deepfake video clips has been collected from YouTube website with HD quality and generated by swapping famous politicians face to the pre-existing video clips.

- iv) Statistical analysis was performed for the collected datasets to analyse the length distribution of the titles, contents and headlines for fake and real news articles, clickbait and non-clickbait headlines.
- v) Word clouds for most frequent occurring words for both real and fake news articles using Tf-Idf and CV feature extraction techniques. Also, the word clouds of clickbait and non-clickbait headlines on all three collected datasets after pre-processing phase (using *GloVe* embeddings) has been analysed.
- vi) A post-processing approach is proposed on the extracted text to analyse the non-verified image quotes. Extraction of real author name from the misquoted images is shown by the system.
- vii) Various borrowed machine learning based models are used to identify the best algorithm for the chosen datasets.
- viii) A novel method is proposed to detect fake news articles by merging the ML models based on false prediction rates. A multi-level voting classifier is proposed to classify the positive and negative instances of articles. The system extracts the textual content from the URL to extract necessary features to identify fake news articles.
- ix) A novel hybrid approach is proposed to detect short clickbait headlines and its types which works under a two-phase structure. In the first phase, the headlines of textual and non-textual data are fed to the CNN-LSTM (with defined layers) model with the use of pre-trained vectors to identify whether the chosen post is legitimate or a clickbait. In the second phase, the classified clickbaits are fed to the existing Biterm Topic Model (BTM) which is a type of short text classifier. It uses a biterm

co-occurrence of words to cluster similar clickbait headlines to identify the types of clickbait. The system extracts the textual content from the URL (clickbait links) or images to identify a clickbait headlines. Also, a graphical analysis has been performed to classify the type of clickbait headlines (among eight categories) using BTM.

- x) A novel approach is proposed to detect deepfake video clips which works under a two-level structure. At first level, the forged frames from deepfake video is extracted using OpenCL and in next phase pre-processing is performed on the extracted frames to feed it to the next level. At second level, an existing temporal-based Convolutional Long Short-Term Memory (C-LSTM) model is used to identify the fake frames to detect the fake face-swap video clips. Merging these two levels gives a novel approach.
- xi) A post-processing approach is proposed on the extracted text to analyse the non-verified image quotes.
- xii) Extraction of textual content from the URL to extract necessary features to identify fake news articles and clickbait headlines.
- xiii) Graphical analysis has been performed to classify the type of clickbait headlines using BTM.
- xiv) User-friendly analysis is performed on the retrieved frames from deepfake video clips on the basis of time interval.
- xv) Design and development of the system for statistical analysis of multi-modal signals (text, video, images, URLs) using Python web framework-Django by designing a generic approach for data-science related cross-platform application for end-users.

- xvi) Public consumption of a RESTful API to identify fake or unverified contents at real-time.

1.10 Thesis Organization

The thesis has been structured into seven chapters. The brief overview about these chapters is given as follows.

Chapter 1 presents the need of fake news detection system by introducing various types of fake contents seen over social media platforms. The chapter also introduces the growth of content over Internet in form of various multimodal signals (images, articles, posts, videos). It includes the motivation, impact of fake content dissemination over social media platforms to the society.

The generic process which includes dataset collection, pre-processing, feature extraction, fake content detection for both textual and non-textual data, output analysis phases are discussed in detail. The chapter concludes with thesis organization, research gaps, research methodology, research objectives, thesis contributions. The remaining chapters of the thesis are based on the objectives.

Chapter 2 elaborates a literature review of fake news detection over social media platforms for different multimodal signals (images, articles, quotes, clickbaits, deepfake videos). The evolution of fake news creation and detection is discussed in this chapter. To conduct literature review, a proper methodology has been followed. The review presented in this chapter has been performed by identifying the related research studies from the renowned

electronic databases as well as the topmost conferences related to the area.

To narrow down the count of selected studies, inclusion-exclusion criteria have been followed. Further, the extracted outcomes are arranged on the basis of their year-wise publications. Different sources for workshops, conferences, journals are used to download the research articles. Also, the online freely datasets available are discussed in this chapter. The different types of existing fake content detection techniques using machine learning and deep learning models are also discussed in detail. The year-wise increase in research area to detect fake content online is represented by graph to help the researchers to know latest research work.

Chapter 3 explains the proposed multi-level voting model to detect fake news articles. Various machine learning (ML) models used to build the voting classifiers using three datasets to train the models using three feature extraction techniques (Tf-Idf, CV, HV) are also discussed in this chapter.

The methodology along with its components followed by the architectural framework of the proposed model to detect the fake news articles has been visualized. The proposed multi-level voting model not only helps to improve the accuracy but also helps to reduce the training time of the classifiers. Discussions are done related to the reduction seen in training time of the proposed model, by introducing parallel methods where the base learners are generated parallelly. The motivation behind the proposed model is also discussed in detail in this chapter. The testing benchmark and result comparison of proposed multi-level voting model with various state-of-the-art models has been also discussed.

Chapter 4 explains the approach to identify the textual content from the

images using the OCR tool and perform post-processing technique to improve the accuracy of the retrieved text. A web-based text similarity is used to obtain the URLs of similar text obtained from the images and verify the extracted domain names against the authentic quotation sites database (own created ground dataset) has been discussed in this chapter. The overall methodology and framework of the proposed two-phase hybrid CNN-LSTM structure to detect the clickbait headlines from textual and non-textual dataset (collected and own created) using Glove embeddings has been explained. Further, Bitern Topic Model (BTM) is discussed which is used for automatic identification of the eight types of clickbait headlines into reasoning, number, reaction, revealing, shocking/unbelievable, hypothesis/guess, questionable, forward referencing. The clustering analysis to detect the quality of clusters and comparative study is performed with the existing systems to show the effectiveness of our proposed model.

Chapter 5 explains the ground benchmark dataset created for deepfake video clips of famous politicians in detail. Different tools used to generate first and second generation deepfake video clips are discussed in this chapter. Also, a two-level structure is proposed which is used to detect the deepfake video clips of politicians. At first level, the forged frames from the video clip is extracted and further pre-processing is performed on the extracted frames to feed it to the second level. A deep temporal-based Convolutional Long Short-Term Memory (C-LSTM) model is used to identify the fake face-swap video clips. Apart from this, the framework to detect deepfake video clips and the architecture of the proposed system is discussed. The comparison analysis is also performed with various state-of-the-art techniques, which shows the efficiency of the proposed model on our collected dataset on the

basis of performance metrics and training time.

Chapter 6 describes overall design and implementation of Progressive Web Application (PWA), to detect fake multimodal signals over social media platforms.

The goal of developing such an application is to outreach the social media users to give the power to analyse the reliable, trustworthy and real content for social development. Various features of the proposed systems are discussed in detail. The system architecture of the developed application which uses Django framework is explained in this chapter.

Four systems have been integrated and their framework (client-end and server-end) has been discussed in this chapter. In System design, on client side HTML, CSS and JavaScript is used and on server side Django framework is used to develop the RESTful APIs. Nginx is used to act as reverse proxy and serve as load balancer. For saving data in DB, PostgreSQL DB is used in a Django application. At the client side, user uploads the inputs like images, videos or links as per the service inside a form.

Chapter 7 concludes the research work presented in this thesis and presents the future scope of the work. It has been concluded that the results given by the system are very promising which can be used for the benefits of the society. Also, various limitations and discussions are covered in this chapter.

The developed system can help in controlling the various types of fraud videos (related to political speeches), the spread of fake content during elections, terrorism, natural calamities, crimes from growing over social media for the betterment of the society. In future, it will become easy for the researchers to classify the category of clickbait, fake news article, deepfake

video at maximum during natural events. Also, the efficiency and efficacy of the proposed models can be improved in future to predict the fake contents over social media platforms.

Chapter 2

Literature Review

The explosive growth in fake content and its erosion to democracy, justice, and public trust has increased the demand for its analysis, detection and intervention. Fake content represent the most popular forms of false and unverified information, respectively, and should be detected as soon as possible for avoiding their dramatic effects. The interest in effective detection techniques has been therefore growing very fast in the last years. This literature comprehensively reviews fake content related research. It focuses on fake news definitions, identifies and specifies various approaches to deal with fake content spread over social media platforms. In this chapter current fake content research is reviewed, summarized and evaluated.

2.1 Evolution of fake content detection analysis

Fake content denotes a type of yellow press which intentionally presents misinformation or hoaxes spreading through both traditional print news media and recent online social media. Fake news has been existing for a long time,

since the “Great moon hoax” published in 1835 [1]. In recent years, due to the booming developments of online social networks, fake content for various commercial and political purposes has been appearing in large numbers and widespread in the online world. With deceptive words, online social network users can get infected by these online fake content easily, which has brought about tremendous effects on the offline society already. During the 2016 U.S. president election, various kinds of fake news about the candidates widely spread in the online social networks, which showed a significant effect on the election results. According to a post-election statistical report [7], online social networks account for more than 41.8% of the fake news data traffic in the election, which is much greater than the data traffic shares of both traditional TV/radio/print medium and online search engines, respectively. An important goal in improving the trustworthiness of information in online social networks is to identify the fake content timely, before being disseminated over social media platforms.

Definition 1: Fake news is false news, where news broadly includes claims, statements, speeches, posts, among other types of information related to public figures and organizations.

Definition 2: Fake news is intentionally and verifiably false news published by a news outlet.

These definition addresses the public’s perception of fake content seen in form of fake news, especially following the 2016 U.S. presidential election [28].

2.2 Methodology

There are four major steps which were followed to conduct the literature review on FCD system.

2.2.1 Review protocol

To conduct a literature review, topmost conferences and electronic databases related to FCD is considered. Further, inclusion-exclusion criteria has been applied to extract the limited count of research articles. In next step, the count of the research studies have been reduced on the basis of formulated research questions and then after performing these analysis, final results are compiled.

2.2.2 Research questions

To perform proper survey of research studies, set of research questions have been formulated as shown in Table 2.1.

With the help of these formulated questions, the review of the related work from the electronic databases has been performed in an efficient form. Various open-source tools, machine learning and deep learning models have been explored which can be used by researchers to identify FCD techniques.

2.2.3 Database sources

To start with the search process, electronic database of journals, conferences and magazines were considered. To identify the research studies, the selected

Table (2.1) Formulation of research question for literature review

Research question	Motivation
What is the year-wise list of publications in electronic databases after the evolution of FCD for multimodal signals over social media platforms ?	Fix the sources of publication and time-frame in which maximum number of research studies are published related to area of our research.
How to know the impact of publications related to research field ?	Use the citation information to identify the impact of research studies.
What type of multimodal signals have been explored by the researcher to detect fake content ?	Explore the textual content (including news articles, blogs, URLs, clickbait headlines) and non-textual content (images, audios, videos).
Which FCD techniques have been mostly used in the existing research studies?	Identify the most commonly used ML and DL models for detecting fake news content.
What important factors were considered by researchers during FCD analysis ?	Identify the characteristics of fake and real content spread over social media platforms.
What are the open-source tools available online to detect fake content ?	Explore the tools available which can perform automatic fake content check.
What are the unexplored areas by researchers related to existing FCD over social media platforms ?	Identify the future indirections from the research studies.

electronic databases are Google Scholar ¹, IEEE Xplore ², ACM Digital Library ³ and Science Direct ⁴. It was observed that most of the papers were covered by Google Scholar. Other research papers were extracted from IEEE Xplore, ACM Digital Library and Science Direct were common with the retrieved results from the Google Scholar. So, these redundant papers were excluded before the final outcome.

2.2.4 Inclusion-exclusion criteria

An advance keyword based search has been followed to extract the relevant research articles within a time frame of 2014-2020 from the electronic-databases as shown in Table 2.2. It was observed that majority of the research papers were covered by Google Scholar. Further, keyword strings were also searched from other electronic databases and the redundant research articles were reduced, *i.e.*, these articles were deleted from IEEE Xplore, Science Direct and ACM Digital Library when found in common with Google Scholar database.

Both quantitative and qualitative studies related to research articles have been considered. Keyword-based search give high count of research articles published in journals, conferences and magazines. The advance string search like “Identify fake content [in] social media” and “Fake news detection [in] social media” has been performed in title for and abstract for Google Scholar, IEEE Xplore, Science Direct, ACM Digital Library. Hence, to include all type of research studies relevant to this area, the string search is also included to

¹www.scholar.google.co.in/

²www.ieeexplore.ieee.org

³www.acm.org/dl

⁴www.sciencedirect.com

Table (2.2) Advance keyword-based search for extracting research articles

Source	Keyword	Publication type	No. of results
Google Scholar	Fake news detection [in/over] social media	C/J/M	507
	Identify fake content [in/over] social media	C/J/M	84
IEEE Xplore	Fake news detection [in/over] social media	C/J	23
	Identify fake content [in/over] social media	C	13
Science Direct	Fake news detection [in/over] social media	J	8
	Identify fake content [in/over] social media	C	1
ACM Digital Library	Fake news detection [in/over] social media	C/J	3
	Identify fake content [in/over] social media	C	2

get efficient results.

The exclusion criterion has been applied at different stages to retrieve the research articles from various conferences, magazines, journals as shown in Figure 2.1.

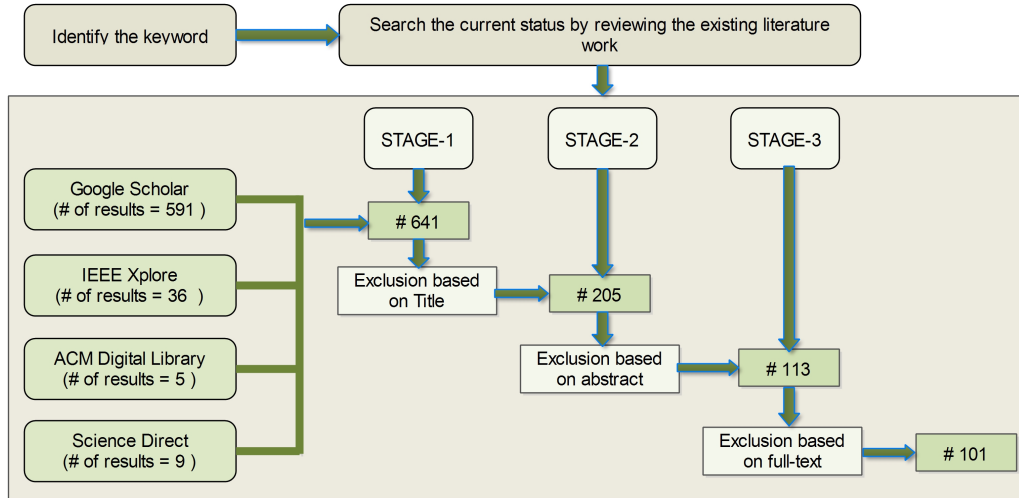


Figure (2.1) Inclusion-exclusion techniques followed for literature review

Our search returned 641 research articles, which were further reduced to 205 on the basis of their titles, and further to 113 on the basis of their abstract and at last to 101 on the basis of full-text length. The retrieved 101

research articles were used for in-depth analysis in literature review.

2.3 Extraction Outcomes

The goal is to identify the year-wise increase in the research studies related to fake content identification over social media platforms. Figure 2.2 represents the source of publications and year-wise status of research studies to answer RQ1, where the year-wise list of publications is evaluated for FCD over social media platforms using multimodal signals.

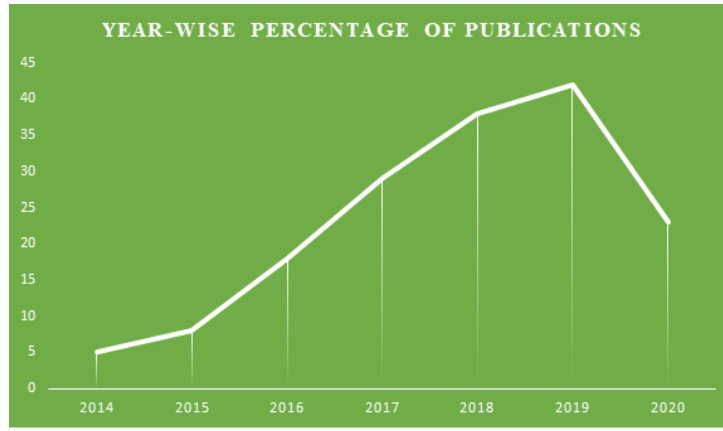


Figure (2.2) Year-wise (*till end of July, 2020) evolution of FND research studies

To address RQ2, three highest cited research articles [7, 29, 21], with more than 100 citations are shown by Table 2.3. These three research articles have

Table (2.3) Count of publications according to citations

Citation count	≤ 5	> 5 and ≤ 25	> 25 and ≤ 120	> 120
No. of publications	48	32	18	3

been marked as a benchmark for other researchers to identify FCD techniques over social media platforms.

2.4 Datasets

A variety of datasets in form of various multimodal signals (text, articles, images, videos, posts) for fake content identification have been published in recent years. The datasets have been categorized into two different categories, textual and non-textual data.

2.4.1 Textual dataset

When comparing fake content datasets, a few trends can be seen. Most of the datasets are small in size, which can be ineffective for current machine learning models that require large quantities of training data. Only four datasets contain over half a million samples, with CREDBANK [17] with *certainly inaccurate, probably inaccurate, uncertain or doubtful, probably accurate, certainly accurate* labels and FakeNewsCorpus [30] being the largest provides more fine-grained labels like (*Fake, Satire, bias, conspiracy, state, junksci, hate, clickbait, unreliable, political and reliable*), both containing millions of samples. In addition, many of the datasets separate their data into a small number of classes, such as fake or true. Datasets such as NELA-GT2018 [31] with *NewsGuard, Pew Research Center, Wikipedia, OpenSources, Media Bias/Fact Check (MBFC), AllSides, BuzzFeed News, Politifact* labels; LIAR [29] with *pants-fire, mostly-true, half-true, true, false, barely-true* labels. While some datasets include data from a variety of categories [32], many contain data from specific areas, such as politics and celebrity gossip [33, 34, 35, 36, 37] as seen in Table 2.4.

Table (2.4) Statistics of existing textual fake content datasets

Dataset	Data type	Source	No. of classes	No. of samples
FA-KES	Syrian war	15 news outlets	2	804
FAKENEWSNET	Political/celebrity	Twitter	2	6,02,659
FakeNewsCorpus	variety	Opensources.co	10	94,00,000
NELA-GT-2018	variety	194 news outlets	8	7,13,000
Breaking!	political	BS Detector	2,3	700
CREDBANK	variety	Twitter	5	6,00,00,000
PHEME	variety	Twitter	2	330
Some-like-it-hoax	scientific/conspiracy	Facebook	2	15,500
BUZZFACE	political	Facebook	4	2,263
BUZZFEEDNEWS	political	Facebook	4	2,282
FEVER	variety	Wikipedia	3	1,85,445
LIAR	political	Politifact	6	12,836

2.4.2 Non-textual dataset

Most of the existing fake content datasets collect only textual data. However, fake data can also come in the form of images or videos.

Existing fake image datasets are limited in size and diversity, making dataset research in this area important. Image features supply models with more data that can help immensely to identify fake images and content that have image data. Traditional fake image datasets that have been published are shown in Table 2.5. The Image Manipulation dataset [38] contains manipulated images for detecting image manipulation. The PSBattles dataset [39] is an image dataset containing manipulated image derivatives from one subreddit. The image-verification-corpus [40], contains both text and image data. Fakeddit dataset uses three way classification approach (2-way, 3-way and 6-way). The first 2-way includes *fake or true* label, 3-way includes *true, fake or fake with false context*, 6-way includes *true, satire/paroday, mislead-*

Table (2.5) Statistics of existing non-textual fake content datasets

Dataset	Data type	Source	No. of classes	No. of samples	Modality
Fakeddit	variety	Reddit	2, 3, 6	10,63,106	text, image
The PS-Battles	manipulated content	Reddit	2	1,02,028	image
Image-verification-corpus	variety	Twitter	2	17,806	text, image
Fauxtography	variety	Snopes, Reuters	2	1,233	text, image
Image Manipulation	variety	self-created	2	48	image

ing content, manipulated content, false connection and imposter content.

Deepfakes (AI-synthesized face-swapping videos) is an emerging problem threatening the trustworthiness of online information. The need to develop and evaluate deepfake detection algorithms required large-scale datasets. In this section, we present a statistical study of large-scale deepfake video datasets as shown in Table 2.6. Also, a comprehensive evaluation of deepfake datasets is done to demonstrate the escalated level of challenges posed by available datasets. First-generation datasets are the datasets that contain less than 1,000 videos. The subjects featured in the videos of these datasets have non-consensual appearance in synthesized videos as most of these videos are sourced directly from YouTube. Second-generation datasets have videos with better perceptual quality and the number of videos in these datasets range between 1 and 10,000 videos with a greater duration of videos than the videos in first generation. Third-generation datasets contain tens of thousands of videos and tens of millions of frames. In addition, these datasets explicitly indicate that they have the rights to the underlying content or consensual agreement from subjects in the dataset.

Table (2.6) Statistics of existing deepfake datasets

Dataset	Deepfake clips		Real clips		Generation	Year of release
	No. of videos	No. of frames	No. of videos	No. of frames		
UADFV	49	17,300	49	17,300	First	2018
DF-TIMIT-LQ	320	34,000	320	34,000	First	2018
DF-TIMIT-HQ	320	34,000	320	34,000	First	2019
FF-DF	1,000	5,09,900	1,000	5,09,900	First	2019
DFD	3,068	22,42,700	363	3,15,400	Second	2019
DFDC	4,113	17,83,300	1,131	4,88,400	Second	2019
Celeb-DF	5,639	21,16,800	590	2,25,400	Second	2019

The UADFV dataset [41] contains 49 real YouTube and 49 deepfake videos. The deepfake videos are generated using the DNN model with FakeAPP [42]. This dataset contains 49 real videos, which were used to create 49 deepfake videos. The average length of these videos is approximately 11.14 seconds, with a typical resolution of 294×500 pixels. The DeepFake-TIMIT dataset [43] includes 640 deepfake videos generated with faceswap-GAN [44] and based on the Vid-TIMIT dataset [45]. The videos are divided into two equal-sized subsets: DF-TIMIT-LQ and DF-TIMIT-HQ, with synthesized faces of size 64×64 and 128×128 pixels, respectively.

The FaceForensics++ dataset (FF-DF) [46] includes a subset of deepfake videos, which has 1,000 real YouTube videos and the same number of synthetic videos generated using faceswap [47]. The Google/Jigsaw deepfake detection dataset (DFD) has 3,068 deepfake videos generated based on 363 original videos of 28 consented individuals of various genders, ages and ethnic groups. The details of the synthesis algorithm are not disclosed, but it is likely to be an improved implementation of the basic deepfake maker algorithm. The Facebook deepfake detection challenge dataset (DFDC) [48] is part of the deepfake detection challenge, which has 4,113 deepfake videos created based on 1,131 original videos of 66 consented individuals of various

genders, ages and ethnic groups. The Celeb-DF [49] dataset is comprised of 590 real videos and 5,639 deepfake videos (corresponding to over two million video frames). The average length of all videos is approximate 13 seconds with the standard frame rate of 30 frame-per-second. The real videos are chosen from publicly available YouTube videos, corresponding to interviews of 59 celebrities with a diverse distribution in their genders, ages, and ethnic groups.

2.5 Open-source tools

Many reliable resources are available through which news articles, stories can be verified. Due to the increase in the significance of the fake content related problem, several solutions have already been proposed and released. They are mostly created as extensions to the browsers. They use different approaches for fake content detection, namely: image, author and textual analysis. In this section, a short overview is provided for the online tools selected and used in this research.

- i) **B.S. Detector:** It is an Open Source which helps to alert the users from unreliable news sources. It works on the basis of searching all links given on a web page for references to untrusted sources. There is a manually compiled list of domains against which the B.S. Detector [50] checks the sources and provides the warning if presence of questionable websites or links is seen. The tool B.S. Detector is a browser extension which supports Chrome, Mozilla, Safari, and Edge.
- ii) **SurfSafe:** The SurfSafe tool [51] is a free browser extension released by RoBhat labs in August 2018 available online. The main idea of this

tool is to compare the images from news to a database of images. The images stored in the database are called from both trusted and fact-checking sites. If the image was modified or used in fake context, the whole news piece is considered to be fake.

- iii) **Fake News Detector AI:** The Fake News Detector AI is a tool available online [52]. There is no information about the algorithms involved for its detection procedure. The online interface provides the information ‘true’, ‘false’ or ‘unknown error’ for each link that the user wants to verify.
- iv) **TrustedNews:** The next tool involved in the research is called TrustedNews and can be found online [53]. It was released by the MetaCert organization in 2017. This tool can provide a wider set of results - trustworthy, untrustworthy, satire, biased, malicious, clickbait, generated and unknown. Trusted News is powered by the MetaCert Protocol, and it is used for independent, politically objective data sources to measure the truthfulness of news content.
- v) **Fake News Detector:** This tool is available online [54]. It involves the feedback provided by the other users of the tool. It can give one either real (legitimate), fake news, clickbait or biased as an output. It is an open source project and its repositories are available on the GitHub platform.
- vi) **Fake News Guard:** This tool is a passive one working as a browser extension [55]. It verifies any page visited by the user and any link displayed in Facebook. However, it is difficult to evaluate the way this tool works. Authors only claim that it combines the linguistic approach,

network analysis and artificial intelligence. Apart of that, each user can report the source, if it is suspected of being fake.

vii) **FiB by Devpost:** It helps to provide an accurate newsfeed. This tool has 1.5 billion users which works according to the two-fold algorithm. First step evolves consumption of the content where the chrome extension goes through Facebook feed in real time. The FiB by Devpost checks the posts authenticity where posts are the links, updates or images. Artificial intelligence helps to check and verify the posts using keyword extraction, source validation techniques. It uses Twitter search to verify the screenshot on Twitter [56] to know whether post is authentic or not. If the post found is false then AI will help to find the true information and will display it. Second step evolves the content creation which is seen at every time a user shares the content. JavaScript is used by chrome extension to extract the links, posts, and images. The plug-in is available on chrome store. The data retrieved is then sent to a collection of API calls, which produces a single trust factor. Back-end of FiB is written in Python.

viii) **PolitiFact:** It is a website which is also available as an android application. This application [57] helps to check the statements which are stated by politicians. The accuracy of the tool is rated on Truth-O-Meter. A list of sources is published with every Truth-O-Meter item according to the recorded interviews. The main motive of this tool is to help the users to judge which ruling party is better for them. The Truth-O-Meter has six ratings which decrease according to their level of truthfulness.

Most of the tools discussed above maintain the database in form of links,

images to match the verifying content. Whereas, other tools detect fake content with the help of feedback received from the users. The existing tools only deal with textual and image-based dataset to verify the content by either string or image matching from the collected dataset. No real-time tool is available to detect the fake video clips, clickbait headlines, quoted images or fake articles together in form of an integrated system.

2.6 Classification of fake content detection techniques

An evaluative summary of information is found through a survey, based on which this section is divided into four sub-sections. The sub-sections are made on basis of the content generated on OSMs and the users who are generating this content.

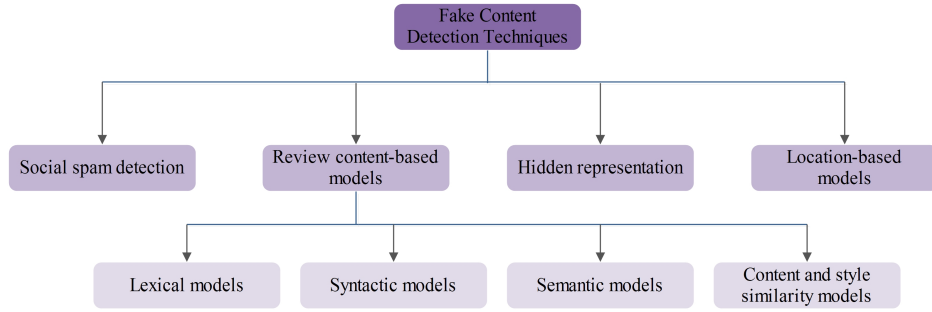


Figure (2.3) Hierarchical classification of fake content detection techniques

Figure 2.3 depicts the hierarchy followed to discuss the fake content detection techniques studied through literature review [58] [59].

2.6.1 Social spam detection

Social framework is a kind of labelling where any client can undoubtedly relate a type of tag to some other client in the framework. This section will introduce social spam detection [60] where spammers have made social bookmarking sites as prime targets. Most of the social sites require energy and time to manually remove or filter spam. Benjamin *et al.* (2009) proposed six highlights of tagging systems catching diverse properties of social spam. Utilizing the six proposed highlights, creators assessed different administered machine learning calculations to identify spam with precision over 98% with a false positive rate of 2%. An open dataset was utilized which was discharged by BibSonomy.org as a piece of test on spam identification on social tagging systems [60]. Weka tool is used to check the best accuracy achieved by AdaBoost.

To address the issue of detecting video promoters and spammers. Fabricio *et al.* (2009) manually assembled test gathering of genuine YouTube clients and classified them as legitimates, spammers and promoters. Then to investigate the feasibility for detecting spammers and promoters, they applied a supervised classification algorithm [61]. Results reveal that the approach used misclassified only a small number of legitimate users whereas identifies majority of promoters correctly.

It is noted that a user will more likely respond to a friend on Facebook than on traditional email, thus making social spam more effective distributor. To challenge this, an initial study is done by Hongyu *et al.* (2010) to characterize and quantify spam campaigns which are launched using accounts on online social media. Authors have introduced an investigation to examine and measure attempts to spread the malicious contents [62] on OSNs.

A spam recognition system is required to recognize suspicious users. One

such model is proposed by Wang (2010), which utilizes a directed social graph to investigate the connections between friends and follower among Twitter. Twitter API techniques are used to build up a web crawler which utilizes 49M follower/friend connections, 25K clients, 500K tweets, all together are collected, which are publicly accessible on Twitter. Bayesian classifier tests are conducted to analyse the dataset and evaluate the execution of a spam detection system giving 89% precision [12].

Another approach to detect spammers on Twitter was given by Fabricio *et al.* (2010), which are used to classify the users as spammers and non-spammers. Using Twitter API, 54M users, 1.9B links and 1.8B tweets were collected. Using the dataset retrieved, authors identified number of characteristics related to tweet content and user social behavior [13]. These characteristics are further used for distinguishing users as spammers and non-spammers. The result highlights that approximately 70% are correctly classified as spammers and 96% of are correctly classified as non-spammers.

Gupta *et al.* (2012) analysed the credibility of information during 14 high impact news events of 2011 through tweets. To predict the credibility of information in a tweet regression analysis was done to identify the source and content-based features [63]. A supervised machine learning algorithm, SVM-ranking based model was built to rank tweets according to credibility score.

Faraz *et al.* (2012) proposed a Markov Clustering approach [64] which is an unsupervised algorithm to detect spam profiles on OSN. This approach is used to generate different clusters for different categories of profiles. Real dataset of Facebook profiles is taken which have both benign and spam profiles. The model developed uses weighted graph where profiles are represented as nodes and the interactions between profiles as edges. The pairs connected

via edges, calculates the real social interactions in terms of page likes, shared URLs, active friends within the network. Table 2.7 shows the models used with various proposed approaches developed by the researchers.

2.6.2 Review content-based models

Researchers have used different techniques to identify and review the fake content. The details of these models have been discussed below.

Lexical models

Bag of Words (BOW) is one of the best common features used as a part of lexical models. This comprises of a group of words retrieved from the text, from which n-gram [1] features can be extracted. The second most important feature is Term Frequency (Tf), which are similar to bag of words approach but are related to frequency of the words.

Kevin *et al.* (2011) designed a novel method to automatically identify and rank social network users according to expertise for any given topic. For the domain of expertise like wine, dating, football, cars, investing, fantasy [10]; authors manually selected ten twitter accounts of high-level expertise from each domain. WeFollow; a popular Twitter directory service was used to collect the accounts from Twitter.

Niall *et al.* (2015) proposed a hybrid approach which combines both machine learning and linguistic cues with network-based behavioral data. The hybrid approach follows both n-gram and Bag of Word techniques [65] to represent data.

Hadeer *et al.* (2017) proposed a fake news detection model which uses n-gram analysis and Term Frequency- Inverse Document Frequency (Tf-Idf)

Table (2.7) Summarized review of research studies related to spam detection techniques

Authors	Proposed approach	Model	Features
Benjamin <i>et al.</i> (2009) [60]	Analysed distinct 6 features for detecting social spammers using Machine Learning.	SVM, AD-aBoost	TagSpam, TagBlur, DomFp, NumAds, Plagiarism, ValidLinks
Fabricio <i>et al.</i> (2009) [61]	A video response crawler is proposed to identify spammers in online video social network.	SVM	Video attribute, Individual characteristics of user behaviour, Social relation between users via video response interactions.
Wang (2010)[12]	Developed a web crawler which relies on API methods provided by twitter.	Bayesian classifier	Content-based, Graph-based
Fabricio <i>et al.</i> (2010) [13]	Identified the characteristics related to user-behavior and tweet content for detecting spammers.	SVM	Content attributes, User behavior
Gupta <i>et al.</i> (2012) [63]	A model is built to find the credibility of information in tweets.	SVM-ranking algorithm	Content attributes, User behavior
Faraz <i>et al.</i> (2012) [64]	A graph-based social network model is proposed to identify spam profiles on Facebook.	MCL	Facebook pages, Wall posts, Tags

as a feature extraction technique. In this paper, six classifiers of machine learning are used and two different feature extraction techniques are used for comparison and investigation. Six supervised classification [1] techniques are K-Nearest Neighbour (K-NN), Support Vector Machine (SVM), Logistic Regression (LR), Linear Space Vector Machine (LSVM), Decision Tree (DT), Stochastic Gradient Descent (SGD). A public dataset from fake and real news websites is taken for experimental evaluation.

Clickbait is used to describe a web content through which an online advertising revenue can be generated by compromising the quality and accuracy of the content. Svitlana *et al.* (2017) built a predictive model to manage 130K news posts as verified or malicious and classified four subtypes of suspicious news [56] such as propaganda, clickbait, hoaxes and satire.

Gabriel *et al.* (2018) did the first analysis of behavior among public groups in WhatsApp regarding political discussions. Dataset was collected over a 28-day period at 3 different layers (message, user, group), over 270K messages and 7K users were retrieved. Many metrics are introduced by authors [9] to characterize the behavior of communication group in WhatsApp. Various types of lexical model techniques are discussed in Table 2.8. Among the various techniques discussed for information retrieval, Tf-Idf is the best technique. Tf-Idf algorithm assigns a weight to the keyword in any content based on the importance of the keyword. To extract the textual features from headlines, news articles, textual posts, TF-IDF feature extraction method can play a significant role. In next section, syntactic models are discussed in detail.

Table (2.8) Summarized review of research studies related to lexical model techniques

Authors	Proposed approach	Model	Features
Kevin <i>et al.</i> (2011) [10]	Designed an algorithm for ranking and identifying twitter users.	Latent Dirichlet Allocation (LDA)	Tf-Idf
Niall <i>et al.</i> (2015) [65]	For the implementation of fake news detection tool, a hybrid approach is proposed.	Linguistic and Network models	Bag of Words, n-gram
Hadeer <i>et al.</i> (2017) [1]	Developed n-gram based classifier to differentiate between true and fake articles.	Linear Support Vector Machine	Tf-Idf
Svitlana <i>et al.</i> (2017) [56]	A predictive model was built to predict four sub-types of suspicious new; satire, hoaxes, clickbait and propaganda.	Linguistic models	Tf-Idf, Doc2Vec

Syntactic models

Numbers of researchers turn to Part of Speech (POS) and Linguistic Inquiry Word Count (LIWC) features combined with N-gram to detect fake surveys. LIWC is a software created to identify emotional and cognitive characteristics in written speech. It depends on a built-in dictionary to arrange target words based on various categories, for example, cries is in sadness. Some of the categories they developed are past tense verb, verbs, overall effect, negative emotion and much more.

Rumors are defined as such statements whose truth value is unverifiable. It is a crucial task to identify rumors on OSM. Vahed *et al.* (2011) explored three features such as content-based, network-based and micro-blog specific memes to correctly identify the rumors [66]. Such features are also used to identify dis-informers or users who endorse a rumor and further tries to spread it. For experiment, authors collected 10,000 manually annotated tweets from twitter and achieved 0.95 in Mean Average Precision (MAP). Satire is a type of deception that intentionally mimics the style and format of journalistic reporting. Prateek *et al.* (2013) build a web service platform known as MultiOSN, which is user specific. Through MultiOSN, a real-time event data is collected from 5 different OSNs; Facebook, Twitter, Google+, YouTube, Flickr [2]. With the help of collected data, authors present real-time analytics and visualizations with features like negative and positive tweets of an event, geographical source of an information, an hour wise plot for number of posts on every social media against time, tag clouds are used to represent most frequently occurring words during such events on OSNs.

Victoria *et al.* (2016) proposed a satire detection model with Support Vector Machine (SVM) based algorithm across 4 domains, such as science, business, soft news and civics [8]. In this paper, two datasets are taken, first

includes 2 satirical and 2 legitimate news sites whereas second includes 6 satirical and 6 legitimate news sites to verify the source of 360 news articles. Five features together are taken which gives the best predicting feature combination with 90% precision and 84% recall to identify satirical news which can help to minimize deception impact of satire. Table 2.9 below gives the summarized review of syntactic techniques used. The machine learning based algorithms like Naïve Bayes, SVM are mostly used for textual analysis of fake or real news content.

Table (2.9) Summarized review of research studies related to syntactic model techniques

Authors	Proposed approach	Model	Source of Data Collection	Features
Prateek <i>et al.</i> (2013) [2]	A web service platform, MultiOSN is developed.	MultiOSN	Facebook, Twitter, YouTube, Google+, Flickr	Sentiment analysis, Geographical analysis, Activity, Tag clouds
Victoria <i>et al.</i> (2016) [8]	A satire detection model is developed.	SVM	US and Canadian national newspapers	Absurdity, Humor, Grammar, Negative affect, Punctuation
Vahed <i>et al.</i> (2011) [66]	Identified tweets in which rumor is endorsed.	Naïve Bayes	Twitter	Content-based, Network-based, Twitter specific memes

Semantic models

Semantic features help with distinguishing similarity of reviews, identifies even if the spammer changes a word in one of their reviews to mislead the readers to think as if it is not the same opinion.

Chen *et al.* (2015) utilized Vector Space Modelling (VSM) and Rhetorical Structure Theory (RST) to analyse misleading and truthful news. RST catches the coherence of story in terms of functional relations among the useful text units and also describe the hierarchical structure for each article/news. VSM is used for identifying the relations among rhetorical structure, *i.e.*, each article content can be depicted as vectors in high dimensional space [67]. Every dimensional space represents a number of rhetorical relations of news reports. Such type of news reports is represented as vectors in n-dimensional space. The two clusters named as, deceptive and truthful news are seen as subsets of news. Each news has distinct features which helps to make all stories unique and identify a story as a part of the cluster.

Aggarwal *et al.* (2018) proposed an unsupervised k-nearest neighbor (K-NN) detection model to identify users who manipulate the follower count. Also, the neighborhood of the users is identified to reflect the user similarity in terms of expected follower count [68]. Using this model, results with 84.2% accuracy are achieved for follower count estimation.

Semantic techniques with existing models used and features extracted are discussed in Table 2.10. Such techniques help to track the similarity evaluations to train the neural network language models.

Table (2.10) Summarized review of research studies related to semantic techniques

Authors	Proposed Approach	Model	Features
Chen <i>et al.</i> (2015) [67]	Analysed coherence relations between deceptive and truthful news.	Vector Space Model (VSM)	Discourse
Aggarwal <i>et al.</i> (2018) [68]	Identification of Temporal signatures to identify customers of black market services.	Unsupervised Nearest Neighbor	Static, Posting behavior, Temporal behavior
Ahmed <i>et al.</i> (2012) [64]	A graph-based social network model is proposed to identify spam profiles on Facebook.	MCL	Facebook pages, Wall posts, Tags

Content and style similarity models

Content and stylometric models are based on stylometric features. Stylometric features can be separated into syntactic and lexical features. Examples of lexical features include the ratio of characters in a word, average sentence length, and the total number of characters. Examples of syntactic features include function words and frequency of punctuations.

User-generated contents are growing tremendously as seen in the past 10 years. So, it is important to analyse the quality of content which is being published online. Magdy *et al.* (2010) made use of web-content to generate a statistical support score for textual documents. An algorithm is proposed where the key facts are retrieved from the content to construct the basic key phrases of the document [69]. A report generated shows a difference between

false facts data and score distance of factual news data.

Sidharth *et al.* (2011) gave major findings regarding phishers who use URL shorteners not only for reducing space but also to hide their own identity. It is being observed by authors that Facebook, Twitter like OSM brands are more targeted by phishers than traditional brands like eBay and HSBC. Also, most of the phishing tweets are spread by attractive words and multiple hashtags from Twitter. This is the first study to connect phishing landscape using blacklisted URLs [70].

Phishing scams are spread with the advent of OSM as phishers use social network like Twitter, Facebook etc. to spread such scams. Over 100 million users post about 200 million tweets every day, to overcome this, a PhishAri technique is developed by Aggarwal *et al.* (2012) to detect phishing on Twitter in real-time. Authors used various twitter features such as the age of account, number of tweets, follower-followee ratio to detect phishing tweets. PhishAri service [71] is deployed as a system for end users by providing a plug-in extension in chrome browser.

Analysis for the real event such as Boston Marathon Bombs is done by Gupta *et al.* (2013) which occurred on 15th April 2013. During the event, it was observed that a lot of fake and malicious profiles originated on Twitter. Results showed that 29% of content originated during the Boston Bombs [72] was viral whereas 51% was generic opinions and comments. Authors identified six thousand profiles which were immediately created after the bombs occurred and was suspended by Twitter.

There are both positive and negative effects of social media regarding real time events. A positive aspect includes the effective disaster management whereas negative aspect includes the malicious entities which are used to spread rumors and fake news. One such event, during Hurricane Sandy

[73] highlights the role of Twitter to spread fake images about the disaster. 10,350 unique tweets consisting of fake images were identified from Twitter using Twitter API. It was observed that 86% of tweets which were spreading fake images on Twitter were retweets. Also, the results showed that top 30 users out of 10,215 resulted in 90% of retweets of fake images whereas only 11% links such as follower relationship on Twitter resulted in spreading fake photo URLs.

Tabloids are often used for exaggeration, sensationalization and producing misleading and low quality news. A new form of tabloidization has emerged known as clickbaiting. There exists both textual and non-textual clickbaiting, which is surveyed by Yimin *et al.* (2015), who proposed a hybrid approach [6] for automatic detection of clickbait.

Shu *et al.* (2017) defined the characteristics of fake news. First characteristic tells that fake news is intentionally written to make readers believe the false information which makes it difficult to detect real news content. Second, the data produced by users involved in social engagements is big, unstructured, incomplete and noisy which are the challenging issues. Authors present a review for detecting fake news on online social media (OSM) including existing algorithms from evaluation metrics, representative datasets and data mining perspective. Also, in this research paper, open problems and future research directions [7] to detect fake news are discussed.

Researchers observed that Facebook pages are generating more malicious content than legitimate ones. Dewan *et al.* (2015,2017) proposed an extensive feature set based on entity profile, textual content, metadata, URL features to identify malicious content at real time on Facebook [74]. The feature set is further used to train multiple machine learning models and achieved an accuracy of 86.9%. This model is used to create restful API. REST API

browser plug-in [75] is used for identifying malicious posts on Facebook in real time. Dataset which was collected and analysed had 4.4M public posts taken from Facebook during 17 specific news making events (natural calamities, sports, terror) over 16 months of period. Dataset was filtered using URL blacklists and human annotations. Authors used 2 independent supervised learning models to identify the malicious posts and achieved over 80% of accuracy.

Wei *et al.* (2017) clearly identified the misleading and ambiguous headlines separately using semi-supervised algorithm [3]. Features used by author to train the model are based on congruence between bodies and headlines of the articles. This model uses Class Sequential Rules (CSR) features for ambiguous headline detection.

Social platforms used in existing research articles along with the proposed approaches developed by researchers are discussed in Table 2.11.

Table (2.11) Summarized review of research studies related to content and style similarity techniques

Authors	Proposed approach	Model	Social Network	Features
Qazvinian <i>et al.</i> (2011) [66]	Identified tweets in which rumor is endorsed.	Naïve Bayes	Twitter	Content-based, Network-based, Twitter specific memes
Aggarwal <i>et al.</i> (2012) [71]	Automatic real-time phishing detection is done.	Naïve Bayes, Decision Tree, Ran- dom Forest	Twitter	URL, WHOIs, Tweet-based, Network-based

Table 2.11 continued from previous page

Gupta <i>et al.</i> (2012) [63]	Analysed number of URLs and @ symbols which are increased during the time of crisis.	LDA	Twitter	User analysis, Content analysis
Gupta <i>et al.</i> (2013) [72]	Analysis of Twitter content during Boston Marathon.	Logistic Regression	-	Tweet content
Gupta <i>et al.</i> (2013) [73]	Identifying fake images on Twitter during Hurricane Sandy.	Decision Tree	Twitter	User-level and Tweet-level
Dewan <i>et al.</i> (2015) [74]	Automatic identification of malicious content on Facebook in real time.	Random Forest	-	Textual content, Entity profile, URLs
Chen <i>et al.</i> (2015) [6]	Build a model to automatically detect clickbait.	SVM and Naïve Bayes	-	Stylometric
Dewan <i>et al.</i> (2017) [75]	An algorithm is proposed to detect malicious posts from WOT reputation score.	SVM, Random Forest	Facebook	User likes, comments and shares
Shu <i>et al.</i> (2017) [7]	Overview of Fake news detection and research directions.	AdaBoost	-	News content, Social context

Table 2.11 continued from previous page

Sen <i>et al.</i> (2018) [76]	Built automated mechanism to detect fake likes on Instagram.	Neural Net-work	Instagram	User attributes, Hypothesis-based
-------------------------------------	--	-----------------	-----------	-----------------------------------

Sen *et al.* (2018) built an automated mechanism to detect fake likes on Instagram. An analysis is done by authors which results into high precision of 83.5% in detecting fake likes based on liking behavior. In this paper, features are detected which contribute towards a genuine like on Instagram [77]. This is the first research done on Instagram platform to detect fake likes.

Various social networking websites as discussed in Table 2.11 can be used to extract the data in form of images, videos, headlines, posts, audios, *etc.* In next section, the approach to detect fake content based on deep processing architectures is discussed.

2.6.3 Hidden representations

In many domains, such as artificial intelligence, bioinformatics and finance, data representation learning is a critical step to facilitate the subsequent classification, retrieval and recommendation tasks. Therefore, a combination of deep processing architectures and automatic feature learning a powerful and a holistic alternative to hand-crafted features.

Ma *et al.* (2016) presents a novel method that continuously learns to identify rumors. The proposed model is based on Recurrent Neural Network (RNN) for learning the hidden features that captures the contextual information of posts over time. Results show that RNN method [78] out performs other hand-crafted rumor detection models and further can be improved via extra hidden layers.

Ruchansky *et al.* (2017) proposed a model called CSI (Capture, Score and Integrate) [21] which combines all three characteristics: text of articles, user responses received, sources users use to promote it for more accurate and automated predictions. The model is based on response and text which uses a RNN to capture temporal patterns of user activity. Experimental analysis reveals that CSI achieves higher accuracy on real world data than existing models. A summary of the proposed approaches related to hidden representation techniques is shown in Table 2.12.

Table (2.12) Summarized review of research studies related to hidden representation techniques

Authors	Proposed Approach	Model	Source
Ma <i>et al.</i> (2016) [78]	A model is proposed to learn hidden representations that capture information of relevant posts over time.	Recurrent Neural Network (RNN)	Twitter and Sina Weibo
Ruchansky <i>et al.</i> (2017) [21]	CSI model is proposed.	RNN (LSTM)	Twitter and Sina Weibo
Wang <i>et.al.</i> (2017) [29]	To integrate metadata with text, a hybrid convolutional neural network is designed.	CNN	Publicly available data from polifact.com

RNNs, when used with added LSTM units inside the Neural Network helps to remember the context from beginning of the input sequence which outperforms simple RNNs with better accuracy.

2.6.4 Location-based models

During last few years, location-based services are giving rise to location-based social network (LSBN). One of the most popular LSBN is Foursquare, which gives incentives to the users who checks-in specific places, for instance mayorships are offered to frequent visitors. In such location-based network where check-ins are shared only with friends, tips and list of mayorships are shared publicly to everyone. Pontes *et al.* (2012) used tips, dones and mayorships in Foursquare [79] with 13 million users to analyse the result which indicates that one can infer home city of around 78% of users among 13 million users within 50 kilometres easily.

Pontes *et al.* (2012) also performed a large-scale study of 3 most popular location-based social networks: Foursquare, Google+, Twitter. Authors used openly geographic information and accessible attributes associated with locatable friends to infer home location. In this paper, experimental results show that it was possible to infer user home city, with an accuracy of 82%, 72% and 67% in case of Twitter, Google+ and Foursquare [80].

In Table 2.13, summarized overview of the existing research articles using different LBSNs.

2.7 FCD analysis for textual data

Many techniques have been developed recently for fake content detection seen in form of fake news. In this section, the closely related research work that has been done on detecting fake textual content over online social media (OSM) is discussed in detail.

Table (2.13) Summarized review of research studies related to location-based techniques

Authors	LBSNs	Analysis	Dataset
Pontes <i>et al.</i> (2012) [79]	Foursquare	Home city was correctly inferred for around 78% of analysed users within 50 kilometres.	Dataset collected from system API of Foursquare
Pontes <i>et al.</i> (2012) [80]	Foursquare, Google+, Twitter	Results infer that up to 60% accuracy is achieved within the radius of 6 kilometres.	Foursquare through system API, Google+ through HTTP requests to the system and Twitter streaming API

2.7.1 Research studies related to fake textual content

Most of the social sites require energy and time to manually remove or filter spam. Markines *et al.* (2009) proposed six highlights (TagSpam, TagBlur, DomFp, NumAds, Plagiarism, ValidLinks) of tagging systems catching diverse properties of social spam [60]. Utilizing the six proposed highlights, creators assessed different administered machine learning techniques to identify spam with precision over 98% with a false positive rate of 2%. The Weka tool used for the experiment gives the best accuracy when evaluated on the basis of Adaboost classifier. To address the issue of detecting video promoters and spammers, Benevenuto *et al.* (2009) manually assembled test gathering of genuine YouTube clients and classified them as legitimates, spammers and promoters. Authors have investigated the feasibility for detecting spammers and promoters by applying a supervised classification algorithm [61]. The

classifier used in this paper correctly identifies the majority of promoters correctly.

Qazvinian *et al.* (2011) explored three features such as content-based, network-based and microblog specific memes to correctly identify the rumors [66]. Such features are also used to identify dis-informers or users who endorse a rumor and further tries to spread it. For the experiment, authors collected 10,000 manually annotated tweets from twitter and achieved 0.95 in Mean Average Precision (MAP). Rubin *et al.* (2016) proposed a satire detection model with Support Vector Machine (SVM) based algorithm across 4 domains, such as science, business, soft news and civics [8]. To verify the sources of news articles, authors have discussed various legitimate and satirical news websites. In this research, five features together are chosen to predict the best predicting feature combination with 90% precision and 84% recall to identify satirical news which can help to minimize deception impact of satire.

Analysis for the real event such as Boston Marathon Blasts is done by Gupta *et al.* (2013). During the event, it was observed that a lot of fake and malicious profiles originated on Twitter. Results showed that 29% of the content originated during the Boston Blasts [72] was viral whereas 51% was generic opinions and comments. Authors identified six thousand profiles which were immediately created after the blasts occurred and were suspended by Twitter. Tabloids are often used for sensationalization, exaggeration, producing misleading and low-quality news content. A new form of tabloidization has emerged known as clickbaiting. There exists both non-textual and textual clickbaiting, which is surveyed by Chen *et al.* (2015), who proposed a hybrid approach [6] for automatic detection of clickbait.

Rubin *et al.* (2015) utilized VSM and RST to analyse misleading and

truthful news. RST catches the coherence of story in terms of functional relations among the useful text units and also describe the hierarchical structure for each article/news. VSM is used for identifying the relations among rhetorical structure [67], *i.e.*, each article content can be depicted as vectors in high dimensional space.

Researchers have used different techniques to identify and review the fake content. One of the best and common feature extraction method is Bag of Words. This comprises of a group of words retrieved from the textual content, from where n-gram [1] features can be extracted. The second most important feature which is similar to the Bag of Words approach is Term Frequency (Tf) which is related to frequency of the words. Conroy *et al.* (2015) proposed a hybrid approach which combines both machine learning and linguistic cues with network-based behavioral data [65]. The hybrid approach follows both n-gram and Bag of Word techniques to represent data. Ahmed *et al.* (2017) proposed a fake news detection system which uses n-gram [69] analysis and Term Frequency- Inverse Document Frequency (Tf-Idf) as a feature extraction technique [1]. Six classifiers of machine learning are used and two different feature extraction techniques are used for comparison and investigation. Volkova *et al.* (2017) built a predictive model to manage 130K news posts as verified or malicious. Authors have classified four subtypes of suspicious news such as propaganda, clickbait, hoaxes and satire [56].

Chhabra *et al.* (2011) had put forward a URL static feature based detection method for detecting malicious websites with accurate results. The author has focussed on external features such as IP addresses. Further, a vector construction VSM [70] is chosen as the URL vector model. The dataset taken consists of malicious URLs which was downloaded from the phishing platform named as 'Phishtank' [71].

Comparative analysis of related work done in the field of fake content detection is shown in Table 2.14.

2.7.2 Research studies related to clickbait headlines

Clickbait has become an imperative subject for research purposes, as it is used by computer research and linguists team. An overview of the related approaches for clickbait headline detection has been discussed in this section.

Vijgen *et al.* (2014) studied listicles which are one of the vital components of clickbaits [81]. The authors deliberated approximately 7,000 clickbaits by Buzzfeed and the shred like “16 Cancer making food you eat every day” or “Do you know 39 celebrities who passed away this year” are some compiled examples from their collected dataset.

A clickbait type known as forward reference was studied by Blom *et al.* (2015) from a Danish news website in the form of headlines and is used for two purposes either to attract the user to click the title or to make the information gap by giving a headline [82]. According to the authors, clickbaits mainly composed of adverbs, articles, demonstrative and personal pronouns.

Ferro *et al.* (2016) brought together approximately 3,000 tweets from the top twenty publishers on twitter among other computer scientists and created a model that used hand-crafted features from three fields which are, the meta information, the linked web page, and the teaser message or title [83]. Among these three, the linked web page comprised of readability and text features, and the teaser message had some basic dictionary and text features, while the meta information had tweets related features. These features were classified into a supervised mechanism and achieved 0.79 AUC-ROC for both precision and recall at 0.76. They also revealed that features retrieved from the first category outperformed rest of the categories taken into consideration, with

Table (2.14) Comparative analysis of research studies related to fake textual data

Authors	Proposed Approach	Model	Dataset	Features
Markines <i>et al.</i> (2009) [60]	Analysed distinct six features for detecting social spammers using machine learning.	SVM, Ad-aBoost	Spam posts, tags	TagSpam, TagBlur, DomFp, NumAds, Plagiarism, ValidLinks
Benevenuto <i>et al.</i> (2009) [61]	A video response crawler is proposed to identify spammers in online video social network.	SVM	Real YouTube user information	Video attributes, individual characteristics of user behavior, social relation between users via video response interactions
Qazinian <i>et al.</i> (2011) [66]	Identified tweets in which rumor is endorsed.	Naïve Bayes	Tweets	Content-based, network-based, Twitter specific memes
Chhabra <i>et al.</i> (2011) [70]	Using URLs static features, a method is developed to detect malicious websites.	Naïve Bayes, Logistic Regression, DT, SVM-RBF, SVM-Linear, SVM-Sigmoid	Malicious URL dataset from 'Phishtank'	Grammar, Lexical, Vectors and Static
Gupta <i>et al.</i> (2013) [72]	Analysis of Twitter content during Boston Marathon.	Logistic Regression	Tweets and corresponding user information	Topic engagement, Global engagement, Social reputation, Likability, Credibility
Chen <i>et al.</i> (2015) [6]	Analysed coherence relations between deceptive and truthful news.	VSM	News samples from NPR's, Bluff the Listener	Discourse
Rubin <i>et al.</i> (2015) [67]	A hybrid approach is proposed combining linguistic and network-based behavior data.	Linguistic, Network models	Simple text sentences	Bag of Words, N-gram
Conroy <i>et al.</i> (2015) [65]	A satire detection model is developed.	SVM	US and Canadian national newspapers	Absurdity, Humor, Grammar, Negative affect, Punctuation.
Ahmed <i>et al.</i> (2017) [69]	Developed n-gram based classifier to differentiate between fake ad real articles.	LinearSVM	News articles	Tf-Idf
Caetano <i>et al.</i> (2018) [9]	A predictive model was built to predict 4 subtypes of suspicious news; satire, hoaxes, clickbait and propaganda.	Linguistic models	News posts	Tf-Idf, Doc2Vec

n-gram and uni-gram features contributing the maximum as they are known for capturing the writing styles.

Eight types of clickbait were expounded by Biyani *et al.* (2016). The authors used the definitions of such types to collect non-clickbait (2,724) and clickbait (1,349) webpages, respectively from the Yahoo homepage [5]. The hand-crafted features like presences of quotes, questions, exclamations, *etc.*, are taken along with traditional features like uni-grams and bi-grams to develop the machine learning model for identifying the clickbait. Comparable features calculated the resemblance between the first five lines of the body and title of the article individually, and non-formality features were used to calculate the quality and formality of pages. Forward reference features were created after had given four types of forward references for clickbait class [82]. They achieved 0.712 precision and 0.548 recall.

A Machine Learning classifier based browser plug-in called ‘Stop Clickbait’ has been proposed by Chakraborty *et al.* (2016). The classifier has been trained by extracting the clickbaits (8,069 articles) which they crawled from web domains such as ViralStories, ScoopwHoop, ViralNova, BuzzFeed and 18,513 Wikinews articles as legitimate posts [4]. They used 14 features spanning to train their SVM classifier and reported the accuracy of 89%, 93% in blocking and detecting the clickbait.

2.8 FCD analysis for non-textual data

Non-textual content seen in the form of videos, images, posts shared over social media platforms is also a matter of concern when dealing with fake contents over such platforms.

2.8.1 Research studies related to unreliable image content

Many domain-specific techniques have been developed to automatically identify fake image using deep learning model recently for fake content detection. In this section, the existing research work that has been done on detecting fake, forged and misquoted images seen over social media platforms have been discussed.

Misquoted images seen over Internet

To identify misquoted images, it is important to first extract textual content embedded on images. Text identification from images is an ongoing research field of pattern recognition. To analyse the issues related to text retrieval of images, various researchers have suggested several technologies, every approach or method attempts to focus on the problems involved differently.

i) Identification of textual-content from images

Gur *et al.* (2012) discuss automated optical character recognition (OCR) tools. It shows that OCR tools do not provide a complete solution and in many cases, human inspection is required for accuracy achievement. The authors have suggested a novel text recognition algorithm based on fuzzy logic rules [84] and its usage, relying on statistical data of the analysed font. This approach enables the recognition of distorted letters that may not be retrieved otherwise, by combining letter statistics and correlation coefficients in a set of fuzzy based rules. The authors focused on calligraphy of the handwritten *Rashi* fonts associated with commentaries of the Bible.

Dutta *et al.* (2012) have proposed a novel research methodology resulting in a 15% decrease in word error rate on highly degraded document images [85] written in Indian languages. AlSalman *et al.* (2012) propose a Braille recognition technique [86], which helps to detect and identify Braille characters embossed on Braille documents. The result is utilized in various applications like embossing, printing, translating, etc. Due to poor quality imaging, the performance of these applications is altered and damaged. The reduced quality imaging is due to many factors like scanner quality, scan resolution, lighting, and various types of embossed documents.

Du *et al.* (2012) proposed Automatic license plate recognition (ALPR) system [87] where the information about the vehicle license plate is extracted from a single image or a series of images. The ALPR uses a color, black and white, or an infrared camera to take images.

An innovative adaptive binarization approach based on wavelet filter [88] has been proposed by Yang *et al.* (2012), showing comparable performance to other similar methods. However, the proposed approach processes faster making it more suitable for real-time processing and easily applicable to mobile devices. This approach makes use of complex scene images of ICDAR 2005 Robust Reading Competition for evaluation.

Bassil *et al.* (2012) proposes a post-processing algorithm for error correction in text detected using OCR tools by using Google's online spelling suggestion [89]. This method helps in detecting and correcting the OCR real-word and non-word errors. The proposed approach reveals a significant improvement in the OCR error correction rate.

Nitrogenous *et al.* (2013) highlights the importance of image binariza-

tion phase in document image analysis and identification [90]. The binarization of an image affects the later stages of document image analysis and identification pipeline. The proposed approach helps in evaluating the images of historical printed/handwritten documents by using pixel-based binarization method.

Manwatkar *et al.* (2015) propose a text recognition method to perform Document Image Analysis (DIA) which is a technique of transforming documents format into an electronically readable format [91]. This method involves scanning of printed documents, converting them to greyscale and binary format, followed by line and then character detection to finally obtain the output text.

Rajan *et al.* (2017) propose a hybrid text detection method from natural scene images. The proposed method uses a fractional Poisson model [92] to improve the quality of the images obtained by Laplacian operation. The enhanced image looks brighter than input and Laplacian images and the image does not contain any noise.

Kushol *et al.* (2018) has proposed a new technique to retrieve text from mobile images using Google Cloud Vision [93]. This experiment gives efficient and effective results in terms of accuracy as well as processing time.

ii) **Text similarity approach to identify unreliable content**

With an enormous amount of textual data available on the web, there is an increasing need to be able to fetch the most suitable document. Estimating the similarity among terms, sentences, statements, and texts is an essential element in numerous tasks like information retrieval, document clustering, word-sense disambiguation, text summarization, *etc.*

Gomaa *et al.* (2013) discusses the existing literature on text similarity [94] by partitioning them into three methods; String-based, Corpus-based and Knowledge-based similarities.

Pradhan *et al.* (2015) highlights using various text similarity approaches like lexical, semantic and fusion similarity [95] for retrieving the most relevant document corresponding to a user's query. The fusion similarity is a hybrid approach with the combination of kernel based similarity and cosine based similarity. Samarinas *et al.* (2018) proposed to generate a question answering system (WAMBy) which extracts answers from top 10 Google search results [96]. A novel text-similarity approach to improve performance of Tf-Idf in synonyms and paraphrase identification has been suggested in [96].

Kowser *et al.* (2019) proposed developing a text similarity method which allows scholars to examine a text by analysing with a regular response recorded in the system. The aim of the system is to reduce the time and also abolish the likelihood of biases using traditional similarity measures like Tf-Idf, Word2Vec, Euclidean [97], Manhattan, Minkowski, Cosine and Jaccard Similarity.

Fake images over social media

A deep-learning based common fake features network (CFFN) is proposed by Hsu *et al.* (2020) to differentiate fake image from real with the use of contrastive loss [98]. At first phase, state-of-the-art GANs are deployed to generate the pairs of fake and real images. At second phase, DenseNet deep network is developed to take pairwise information as the input. At third phase, the proposed network is trained to analyse the features of real and

fake images. At final phase, a classification layer is integrated with the previous phases to detect the input image as real or fake. Authors have used CelebA dataset to produce the training set of fake images using DCGAN, WGAP, WGAP-GP, LSGAN, PGGAN networks.

An automated benchmark is proposed by Rossler *et al.* (2019) for facial manipulation detection [99]. Authors have used *NeuralTextures*, *FaceSWap*, *Face2Face* and *DeepFake* tools to create the benchmark with random size and compression levels. The benchmark consists of 1.8 million manipulated images. They have used domain-specific knowledge which improves the accuracy of forgery detection systems.

Nowadays, Generative Adversarial Network (GAN) are used to generate fake images with realistic appearance. In this research, Hsu *et al.* (2018) have proposed a deep forgery discrimination (DeepFD) to effectively and efficiently identify computer generated images using contrastive loss [100]. The training samples are generated using CelebA dataset. The proposed DeepFD model gives an accuracy of 94.7% to detect fake images generated by GANs.

Tariq *et al.* (2018) detects both human and machine created fake face images [26]. To detect computer generated images through GANs, an ensemble-based neural network (NN) classifier is proposed, which gives 94-99% AUC-ROC score. Also, a neural network model using noise filtering and face-cropping is developed to detect human generated fake face images with 74.9% of AUC-ROC score. To validate the proposed model CelebA and PGGAN datasets are used.

Hashmi *et al.* (2014) has proposed a new approach for copy-move forgery detection in images [101]. Authors have used DWT for decomposing the image into four sub-blocks, *i.e.*, HH, HL, LH and LL. The LL sub-image

is used as it contains more information. Further, the SURF is applied to extract the boundary features from an image. In second approach, authors have combined SURF with DyWT. The results show that when SURF is combined with DyWT, better performance is analysed as DyWT is a shift invariant and does not samples down the information.

Zhang *et al.* (2010) has proposed a approach of forged region on the basis of image edge detection [102]. Firstly, the smoothing filter is applied, then Laplace-Gaussian operator (LOG) edge detection is used to analyse the missing edges. The proposed approach gives efficient results to accurately trace the blur operation located on forged images.

Hsu *et al.* (2012) have proposed a new forgery detection system, which uses novel feature extraction techniques [103]. Gabor filter is used to detect the forged images created by move and copy operations. Various Gabor features representations of an image with different frequencies, rotation angles and scaling factors are generated. The proposed algorithm can locate scale factors, rotation angles and duplicated regions of the forged images. The proposed approach is evaluate on MICC-F220 and Caltech 256 showing that the approach has reliability and robustness properties.

A copy-rotate move (CRM) forgery detection method is proposed by Ryu *et al.* (2010) to analyse the duplicate regions with the help of Zernike moments [104]. The proposed scheme is also resilient to various distortions like blurring, JPEG compression and Gaussian noise. Experimental results show that the proposed approach can accurately localize and identify the CRM regions. One major drawback of the proposed method is that, it is still weak against Affine transform and scaling attacks.

Image description can play an important role to classify fake or forged images. The process requires the detection of associated attributes, relation-

ships, object recognition and people in image. A novel region-based deep learning model is proposed by Kinghor *et al.* (2018) for image description generator [105]. The model uses an encoder-decoder language generator, a regional object detector along with two recurrent neural networks to produce detailed and refined descriptions of a given image. The proposed approach has been evaluated with IAPR TC-12 dataset and outperforms many other state-of-the-art methods when compared using various performance metrics.

Tan *et al.* (2019) proposed a phrase-based image captioning model using a hierarchical LSTM architecture known as phi-LSTM to generate image description [106]. The general state-of-the-art solutions generate the image caption in a sequential manner whereas the phi-LSTM decodes the image captioning from phrase to sentences. The proposed approach is evaluated on MS-COCO, Flickr8k, Flickr30k datasets and gives better competitive results when compared with other state-of-the-art models.

A generative model is proposed by Vinyals *et al.* (2015), which consists of Convolutional neural network (CNN) followed by deep recurrent architecture (a language generator) [107]. The model generates complete natural language sentences from the input image. The model gives high performance when evaluated with other state-of-the-art models on the basis of BLEU-1 score. The proposed approach yields 59, 66, 28, 27.7 BLEU-1 score values when evaluated on Pascal, Flickr30k, SBU and COCO datasets, which is much better than other state-of-the-art models. Comparative analysis of related work done in the field of fake image detection is shown in Table 2.15.

Table (2.15) Comparative analysis of research studies related to fake images seen over social media

Author	Proposed Approach	Model	Dataset	Analysis
Hsu <i>et al.</i> (2020) [98]	A novel CFFN with improved DenseNet is proposed to detect fake images.	CNN	CelebA for training fake images	The precision and recall values obtained for the proposed model on WGAN-GP is 0.986, 0.751; for WGAN is 0.988, 0.927; for LSGAN is 0.947, 0.986; for PGGAN is 0.988, 0.948, respectively.
Hsu <i>et al.</i> (2018) [100]	A novel DeepFD is proposed to identify computer generated images.	CNN	To generate training samples, CelebA dataset is used	The precision and recall values obtained for the proposed model on LSGAN is 0.947, 0.922; for DCGAN is 0.871, 0.844; for WGAN is 0.838, 0.847; for WGAN-GP is 0.818, 0.835; for PGGAN is 0.926, 0.918, respectively.
Tariq <i>et al.</i> (2018) [26]	Two different neural network models are proposed to detect both human and computer generated fake face images.	MTCNN, YOLO, CNN	CelebA and PGGAN	The ensemble shallowNet gives 93.99, 99.99, 99.99, 99.99 AUC-ROC score for 64x64, 128x128, 256x256, 1024x1024 image size.

Table 2.15 continued from previous page

Hashmi <i>et al.</i> (2014) [101]	A series of algorithm is proposed to extract the entire features of the image rather than extracting it from image blocks.	SURF, DWT + SURF, DyWT + SURF	-	The proposed techniques performs better in terms of rotation, invariance, scaling and computational complexity. The true positive rate (recall value) is 64% for DWT +SURF and 76% for DyWT.
Zhang <i>et al.</i> (2014) [102]	A localization approach is proposed which uses LOG edge detector to analyze the forgery in the images.	LOG	Two image sets are taken from baidu website. Image1 has size of 500x375 and Image2 has size of 1024x1024	The forgery is significant in an image when the connected forged region size is detected $\geq 0.85\%$.

Table 2.15 continued from previous page

Hsu <i>et al.</i> (2012) [103]	A novel feature extraction and located algorithm to build a new forgery detection system is proposed.	Gabor filter	The proposed algorithm is evaluated on MICC-F220 and Caltech 256	The proposed approach when implemented for copy-move, rotation, scaling range [0.098, 1.1], scaling range [0.4, 3] and combined rotation with scaling gives 98.86%, 92%, 97.5%, 89.01%, 91% accuracy, respectively.
Ryu <i>et al.</i> (2010) [104]	A copy-rotate-move (CRM) detection scheme is proposed for forgery images.	Zernike moments	12 TIFF images, 100 images from national geographic	F1-score, recall and precision performance metrics are used to measure the suspected regions of images. The average detection rate for CRM for 12 images gives 0.8359 precision, 0.7663 recall and 0.7889 F1-score values.

2.8.2 Research studies related to deepfake clips

Many approaches have been proposed since last decade for the target face manipulations in both images and videos which are known as deepfakes. CNN and various feature extraction methods have been studied in the literature to detect deepfake videos and images.

A universal forensic approach has been proposed by Bayar and Stamm

(2016) to detect image editing operations using deep learning [108]. A new form of the convolutional layer is designed to learn the manipulated features and to suppress an image’s content. The proposed model was tested on the collected dataset (from 12 different models) creating a set of greyscale images. The model gave an accuracy of 99.10% for multi-class classification (detected four different types of forgery) and gave an accuracy of 99.31% for binary classification.

A two-stream (face classification and patch triplet) network for face tampering detection has been proposed by Zhou *et al.* (2017). A CNN model is trained during face classification stream to classify the face images as authentic or tampered. Steganalysis features are used to train second stream for an effective image splicing detection and is used to capture the hidden information from an image [109]. The proposed model was evaluated on the newly collected dataset using FaceSwap and SwapMe tools.

Korshunova *et al.* (2017) studies the problem of transforming the face identity of the source image into the target image using face-swapping. The transformation is performed by preserving lightning, facial expressions and position of the identity in images, with a goal to render the image style of source into destination image. The authors have used CNN to capture the appearance of the target identity using CelebA dataset and have come up with new loss functions to produce high photorealistic results [110].

The realistic fake face videos are developed by deep generative networks with high efficiency and quality. Li *et al.* (2018) have proposed a method using neural networks to expose fake face videos on the basis of the eye-blinking feature. The first step involves the detection of faces in each frame of the video. In next step, the detected faces are aligned into the same coordinate system to discount the changes in orientations of facial landmark

points and head movements [111]. Then, the eye-blinking is detected in each frame of video using the Long-term Recurrent Convolutional Neural Network (LRCN).

It is becoming easier to create face-swaps in videos with a machine learning based freely available software. Such scenarios cause fake terrorism events, political distress, blackmailing unknowns, *etc.* A temporal-aware pipeline based system is proposed by Guera and Delp (2018) to automatically detect deepfakes videos. Frame-based features are extracted by the proposed system using CNN [112]. The retrieved features by CNN model are then fed to RNN to classify the manipulated video within a timespan of 2 seconds.

A capsule network is proposed by Nguyen *et al.* (2019) to detect forged videos and images in a wide range of GANs generated video or image detection. The proposed network consists of five capsules, three as primary and two as secondary or output capsules. Where, the secondary capsules are used to predict the final output (real or fake). The method uses deep convolutional networks and gives promising results for all the four datasets used in their research [113].

A new method is proposed by Yang *et al.* (2019) to detect AI-based generated fake videos and images. Authors have compared the head poses of the identity using only the central regions and all landmarks of the face in images and videos. They used the differences of the established head poses as a feature vector to train the binary classifier (SVM) to detect deepfakes and original images or videos [41]. The results reveal that the SVM model achieves an Area Under ROC (AUC) of 0.843 and 0.89 on DARPA and UADFV corpora, respectively. Table 2.16 shows the existing studies related to the deepfake detection techniques.

Table (2.16) Comparative analysis of research studies related to deepfake detection

Authors	Proposed Approach	Models	Dataset	Analysis
Bayar <i>et al.</i> (2016) [108]	A universal forensic approach has been proposed to detect multiple image manipulations using deep learning.	CNN + RNN	Created own database of edited and unaltered images using 12 different camera models	The proposed approach gives an accuracy of 99.10% to automatically detect the multiple image manipulations.
Zhou <i>et al.</i> (2017) [109]	A two-stream network has been proposed by the author for tampered face detection.	CNN (Face classification stream) + SVM (Patch Triplet stream)	GoogleNet dataset using Steganalysis feature extraction technique for Triplet stream	The proposed approach learns both hidden noise residual features and tampering artifacts.
Guera <i>et al.</i> (2018) [112]	An automatic detection system is proposed for deepfake videos based on the temporal-aware pipeline.	CNN + RNN	Deepfake videp collection from multiple video websites.	A video is predicted as a subject to manipulation or not within 2 seconds of temporal frames.

Table 2.16 continued from previous page

Li <i>et al.</i> (2018) [111]	A new method is proposed to expose fake face videos based on eye-blinking generated by neural network.	LRCN	Created own database Eye Blinking Video	LRCN gave 0.99 ROC curve.
Li <i>et al.</i> (2019) [114]	A method is proposed to detect the difference between deep neural generated images and real scene images by analysing the disparities in colour components.	LDA by extracting various features from the colour components	CelebA, HQ-CelebA and LFW datasets that contains various face images with different resolutions	Average accuracy achieved is > 99%.
Yang <i>et al.</i> (2019) [41]	A method is proposed to detect AI-generated videos or images	SVM using the difference between the estimated 3D head poses as features.	Fake video data (UADFV) Fake image data (DARPA Medial for GAN image or video challenge)	SVM gave 0.89 AUC on UADFV and 0.843 AUC on DARPA corpora.

Table 2.16 continued from previous page

Nguyen <i>et al.</i> (2019) [113]	A novel capsule network with random noise is proposed to detect image and video forgeries.	CNN	Faceforensic, Deepfake, REPLAY-ATTACK, Computer generated images and Photographic images	Capsule random noise network gave an accuracy of 95.93% and 99.23% for image and video Deepfake dataset and an accuracy of 99.37% and 99.33% for image and video Faceforensic dataset with no compression, respectively. Also, the model gave an accuracy of 97% for CGIs and PIs dataset, and give HTER of 0 % for REPLAY-ATTACK dataset.
--	--	-----	--	--

Further, the deepfake study has been divided into two categories to identify deepfake video, images and audio clips.

Machine learning based techniques for deepfake detection

A summary and detailed review of the existing deepfake detection methods using machine learning based approaches are discussed in Table 2.17 and Table 2.18.

Table (2.17) Statistics of existing dataset to detect deepfake using machine learning-based approach

Author	Modality	Dataset	Model	I/P resolution	Statistics of dataset
Yang <i>et al.</i> (2019) [115]	Images + Videos	UADFV, DARPA	SVM with RBF kernel	294×500	49 real and deepfake videos with average length of atleast 11.14 sec
Agarwal <i>et al.</i> (2019) [18]	Videos	Customized dataset	SVM	-	Public speech, news interview related videos are taken into consideration with 10 sec of segment in length using mp4 format and saving 30 fps.
Agarwal <i>et al.</i> (2017) [116]	Images + videos	Created novel video face swapped dataset	SVM	296×296	129 videos are captured from 110 individuals using front cameras of mobile phone with an average video length of 6 sec.
Zhang <i>et al.</i> (2017) [117]	Images	Created own dataset using LFW face database	SVM, Random Forest, MLP	250×250	Total 13,223 color face images are taken. Randomly 5000 swapped faces are generated and 5,000 innocent face images are selected.
Akhtar <i>et al.</i> (2019) [118]	Images + videos	DeepfakeTIMIT	SVM	200×200	320 genuine and 640 manipulated videos from 32 different subjects are used.
Durall <i>et al.</i> (2019) [119]	Images	Created own Faces-HQ dataset	SVM, Logistic Regression, K-Means clustering	1024×1024	20,000 real and 20,000 fake high quality images are taken to perform the experiment.
Korshunov <i>et al.</i> (2018) [43]	Videos	VidTIMIT	SVM	64×64 with LQ, 128×128 with HQ	Total 43 subjects are taken into consideration with 20 videos each from every subject.

Table (2.18) Comparative analysis of existing approaches to detect deepfake using machine learning based approach

Author	Type	Feature	Proposed approach	Performance		
				Accuracy	EER	AUC
Yang <i>et al.</i> (2019) [115]	Reenactment + Replace- ment	Facial land- marks	A new approach based on synthesis models introducing errors from 3D head poses in real and fake faces are estimated.	-	-	0.97
Agarwal <i>et al.</i> (2019) [18]	Reenactment + Replace- ment	2D and 3D facial actions, facial landmark positions, head pose and eye gaze	A one-class SVM is proposed to distinguish between an individual from another deepfake impersonators.	-	-	0.98

Table 2.18 continued from previous page

Agarwal <i>et al.</i> (2017) [116]	Replacement	Facial features like eye, mouth and face-boundary	A novel database named as ‘Swapped’ is created. Also, a novel feature descriptor named as ‘Weighted Local Magnitude Pattern’ is proposed on the basis of attack detection algorithm.	-	18.2	-
Zhang <i>et al.</i> (2017) [117]	Replacement	83 facial landmarks, SURF descriptor, BoW	Authors address the importance for automatic detection of face-swapped images using ML based models.	92.9	94.1	-
Akhtar <i>et al.</i> (2019) [118]	Reenactment + Replacement	LBP, FDLBP, QLRBP, BGP, LPQ, BSIF, CEN-TRIST, PHOG, SIFT, SURF	Author studies different approach to identify the tampered faces in images and videos via local feature descriptors.	-	13.33% for BGP and 17.6% for LBP	-

Table 2.18 continued from previous page

Durall <i>et al.</i> (2019) [119]	Replacement	DFT ap- proach for extracting features from the images	A novel classification approach is proposed for automatic detection of fake images on the ba- sis of frequency domain analysis.	100% for SVM and Logis- ticRe- gres- sion, 96% for K- Means clus- tering	-	-
Korshunov <i>et al.</i> (2018) [43]	Reenactment + Replace- ment	IQM fea- tures	Vulnerability analysis of facenet and VGG based recognition system is done in this work.	-	3.33 for LQ, 8.97 for HQ	-

Deep learning based techniques for deepfake detection

A summary and detailed review of the existing deepfake detection methods using deep learning based approaches are discussed in Table 2.19 and Table 2.20.

Table (2.19) Statistics of existing dataset to detect deepfake using deep learning based approach

Author	Modality	Datasets	Model	I/P resolution	Statistics of dataset
Mo <i>et al.</i> (2018) [120]	Images	CelebA-HQ for real images and created own fake image dataset	CNN	256 * 256	Total 30000 for real and 30,000 for fake images with lossy JPEG com- pression with 95 quality factor.
Li <i>et al.</i> (2018) [111]	Videos	CEW image dataset and created EBV eye blinking video dataset	LRCN	224 * 224	1,193 images of closed eyes and 1,232 images of open eyes from CEW dataset. Downloaded 50 videos with 30 seconds with atleast one eye blink.
Nguyen <i>et al.</i> (2019) [113]	Images + Videos	FaceForensics	Capsule- based CNN	128 * 128	-
Bao <i>et al.</i> (2018) [121]	Images	MS-Celeb-IM	GAN	128 * 128	The dataset consists of five million images of 80,000 celebrities.

Table 2.19 continued from previous page

Do <i>et al.</i> (2018) [122]	Images	Celeb-A	DCGAN	64 * 64	Total 400 images with 200 real and 200 fake are considered. 64 * 64, 256 * 256, 1024 * 1024 sizes of the image were available.
Güera <i>et al.</i> (2018) [112]	Videos	HOHA dataset and collected own deepfake videos from multiple video hosting websites	CNN + RNN	299 * 299	Collected 300 videos from video hosting websites and 300 videos from HOHA dataset. Among which 300 are deepfake and real video clips.
Marra <i>et al.</i> (2018) [123]	Images	Created own image-based dataset	CNN	256 * 256	Total 36,302 images are used with 50 % real and fake categories from apple2orange (4028), horse2zebra (4802), etc.
Cozzolino <i>et al.</i> (2018) [124]	Images + Videos	Celeb-HQ, Celeb-A, created own 20,000 inpainted images, FaceForensics deepfake dataset	CNN-Autoencoder	256 * 256	A total 5 dataset are build with 30,000 images. Total 1,004 fake and real videos are used and are compressed with H.264 quantization parameter.

Table 2.19 continued from previous page

Afchar <i>et al.</i> (2018) [125]	Videos	FaceForensics, and created own dataset from deepfake and Face2Face technique	CNN	256 * 256	Total 175 forged videos have been collected from various platforms with minimum 854 * 480 pix- els resolution with H.264 quantization parameter.
Sabir <i>et al.</i> (2019) [126]	Videos	FaceForensics++	CNN- LSTM	224 * 224	Total 1,000 video sam- ples. Among which 720 video clips are used for training, 140 for testing and last 140 for valida- tion.
Nguyen <i>et al.</i> (2019) [127]	Images + Videos	FaceForensics and FaceForen- sics++	Autoencoder with an encoder and a Y-shaped decoder	256 * 256	FaceForensics consists of 1,004 real videos col- lected from YouTube with two sub-divided datasets, i.e., self-reenactment and source to target reenact- ment.
Ding <i>et al.</i> (2020) [128]	Images + Videos	Self created dataset	CNN	-	Total 420,053 images are extracted for 86 celebri- ties. Each celebrity has 1,000 original images ap- proximately.

Table 2.19 continued from previous page

Du <i>et al.</i> (2019) [129]	Images + Videos	FaceForensics++	CNN, AE, GAN	256 * 256	The videos taken are compressed using H.264 with quantization parameter set as 23 for all 1,000 videos. 200 frames per video is used for training and 10 frame per video is used for testing.
Stehouwer <i>et al.</i> (2019) [130]	Images + Videos	Conducted DFFD dataset	CNN + Attention	299 * 299	0.8 million real and 1.8 million fake faces are generated. One frame per second is extracted to reduce the size of DFFD dataset.
Fernando <i>et al.</i> (2019) [131]	Images + Videos	FaceForensics and FaceForen- sics++, FFW	CNN + HMN	224 * 224	For FaceForensic++, 1,000 original and 3,000 manipulated videos; for FaceForensics, 1,040 videos are taken with 480p resolution.

Table 2.19 continued from previous page

Li <i>et al.</i> (2019) [132]	Images + Videos	FaceForensics++	Xception, MesoIn- ception, Unet, UGG16, FN3	256 * 256	Various categories of images from pristine, deepfakes, face2face, faceswap, neuraltextures tool are considered.
Xuan <i>et al.</i> (2019) [133]	Images + Videos	CelebA-HQ; DCGAN, WGAN-GP and PGGAN gener- ated images	CNN	128 * 128	CelebA-HQ dataset is used for real images and for fake image dataset DCGAN, WGAN-GP and PGGAN are used.
Tu <i>et al.</i> (2019) [134]	Images + Videos	MSU-MFSD, Replay-Attack	CNN	224 * 224	MSU-MFSD consists of 35 subjects with 280 videos with 70 real and 210 fake videos. Whereas Replay-attack consists of 1,200 videos, 200 real and 1,000 fake videos.
Tariq <i>et al.</i> (2018) [26]	Images	CelebA, PG- GAN	CNN	1024 * 1024	200k celebrity images are taken from CelebA dataset and 100k GAN generated fake images are taken into consideration.

Table 2.19 continued from previous page

Ciftci <i>et al.</i> (2019) [135]	Videos	FaceForensics, self created	CNN	128 * 128	140 online videos were collected in the wild, i.e., independent of content, context, resolution and compression unlike other existing datasets.
Li <i>et al.</i> (2018) [136]	Images + Videos	DeepfakeTIMIT, UADFV	CNN	64 * 64 for LQ, 128 * 128 for HQ	UADFV contains 98 videos having 49 real and 49 fake video clips. For DeepfakeTIMIT both LQ and HQ videos are taken among which 10,537 are used.
Amerini <i>et al.</i> (2019) [137]	Videos	Faceforensics++	CNN	224 * 224	1,000 original video sequences with three manipulation methods like faceswap, face2face, deepfake are considered.

Table 2.19 continued from previous page

Korshunov <i>et al.</i> (2018) [138]	Videos + Audios	VidTIMIT, AMI, GRID	LSTM	-	VidTIMIT consisting of 10 videos of 3 sec video per person, from GRID corpus 1,000 videos of 3 sec video per person, from AMI 977 closeup speech videos of a person is taken into consideration.
Korshunov <i>et al.</i> (2019) [139]	Videos + Audios	Created own dataset using VidTIMIT, AMI and GRID	LSTM-DNN	-	AMI videos have profile faces where subjects are having free conversation in an uncontrolled environment.
Chan <i>et al.</i> (2019) [140]	Videos	Created own dataset	CNN	256 * 256	62 subject set of 1920 * 1080 resolution of dancing videos were collected from YouTube website where subject is dancing in front of camera for at least 3 minutes.
Yu <i>et al.</i> (2019) [141]	Images	CelebA human face dataset, LSUN bedroom scene dataset	CNN	128 * 128	Total 20,000 real world RGB images are considered

Table 2.19 continued from previous page

Wang <i>et al.</i> (2019) [142]	Images + Videos	FaceForensics++, Celeb-A, FFHQ, DFDC, CelebDF	SVM + VGGnet	224 * 224	High quality fake face dataset are taken from GAN generated images (like STGAN, StyleGAN2) and real data from DFDC announced during Facebook challenge.
Li <i>et al.</i> (2019) [143]	Images + Videos	DFFD, Face- Forensics++, CelebDF	CNN	64 * 64	1,000 real videos from FaceForensic++ have been manipulated with DF, F2F, FS and NT tool to generate fake images. Total deepfake dataset includes 408 real and 795 manipulated face images.

Table (2.20) Comparative analysis of existing approaches to detect deepfake using deep learning-based approach

Author	Type	Feature	Proposed approach	Performance		
				Accuracy	EER	AUC
Mo <i>et al.</i> (2018) [120]	Reenactment + Replace- ment	To identify fake face in- cluding eyes, nose and mouth	Firstly, authors have proposed a CNN-based model to identify fake face images, then have provided the exper- imental evidence to proof the efficiency of their model.	99.4%	-	-
Li <i>et al.</i> (2018) [111]	Reenactment + Replace- ment	Eye land- marks, upper and lower eyelids	A new method is described to detect the synthesized fake videos on the basis of eye-blinking features.	-	-	0.99
Nguyen <i>et al.</i> (2019) [113]	Reenactment + Replace- ment	VGG-19 net- work is used to extract the latent features	A capsule network is proposed using CNNs to detect various kinds of spoofs from recorded videos and images.	99.3%	-	-

Table 2.20 continued from previous page

Bao <i>et al.</i> (2018) [121]	Replacement	Pose, emotion, illumination and background	An open-set identity preserving GAN framework is proposed for disentangling the face attributes synthesized faces from recombined identities.	92%	-	-
Do <i>et al.</i> (2018) [122]	Reenactment + Replacement	Face-based features are extracted	Authors present GAN-based forensic face detection system using PG-GAN and DC-GAN	80	-	0.807
Güera <i>et al.</i> (2018) [112]	Reenactment + Replacement	Used InceptionV3 for feature extraction from frames of the face-videos	A temporal-based pipeline is proposed for automatic identification of deepfake video clips.	-	-	0.81

Table 2.20 continued from previous page

Marra <i>et al.</i> (2018) [123]	Reenactment + Replace- ment	-	Various image forgery detector systems are studied for image to image translation with and without compression.	95% without compression and 89% with compression.	-	-
Cozzolino <i>et al.</i> (2018) [124]	Reenactment + Replace- ment	VGG-16 net is used for image feature extraction	A novel autoencoder-based neural network is proposed, which is capable of analysing the difference between different manipulation domains.	85%	-	-
Afchar <i>et al.</i> (2018) [125]	Reenactment + Replace- ment	Face landmarks	The authors follow a deep learning approach and presents two networks to show the mesoscopic properties of images.	-	-	0.99

Table 2.20 continued from previous page

Sabir <i>et al.</i> (2019) [126]	Reenactment + Replace- ment	XceptionNet models with pre-defined feature ex- traction layer is used to extract the facial features	A two-step approach is followed. In first step, cropping and alignment os faces from the video clips is done. In next step, classification is performed to analyse the manipulated facial regions.	96.9%	-	-
Nguyen <i>et al.</i> (2019) [127]	Reenactment + Replace- ment	Y-shape de- coder is used to extract the facial features	A multi-task learning approach is proposed to perform segmenta- tion and classification simultaneously for de- tecting facial manipu- lation over images.	92.8%	8.18	-
Ding <i>et al.</i> (2020) [128]	Replacement	ResNet-18 is used to extract low- dimensional feature facial images	A deep learning model is build using transfer learning for detecting swapped faces.	98.5%	-	-

Table 2.20 continued from previous page

Du <i>et al.</i> (2019) [129]	Reenactment + Replace- ment	VGGNet is used to map the input image to the feature space	A forgery detection method is proposed named as LAE, which helps to make pre- dictions of correct evidence to boost the generalization accuracy.	92.9%	-	-
Stehouwer <i>et al.</i> (2019) [130]	Reenactment + Replace- ment	CNN model is used to discover more discrimina- tive features	To improve the clas- sification performance to detect manipulated facial regions, a novel attention-based layer is added. Also, a novel metric named as IINC is introduced by the authors.	-	3.11	0.99

Table 2.20 continued from previous page

Fernando <i>et al.</i> (2019) [131]	Reenactment + Replace- ment	Viola-Jones face detector is used to detect the faces. ResNet is used for extracting features	A novel neural mem- ory architecture is proposed for hier- archical semantic embedding prediction and propagation. Also, an adversarial learning paradigm is introduced to train the novel architecture.	99.4%	-	-
Li <i>et al.</i> (2019) [132]	Reenactment + Replace- ment	Facial-based features are extracted	Pixel-level analysis are performed by evaluating various architectures to create a strong baseline of a problem.	98.1%	-	-
Xuan <i>et al.</i> (2019) [133]	Reenactment + Replace- ment	Image-based features are extracted	The forensic CNN- based model is forced to learn more intrinsic features to give better performance to iden- tify real and face face images.	94.7%	-	-

Table 2.20 continued from previous page

Tu <i>et al.</i> (2019) [134]	Reenactment + Replace- ment	Three meth- ods LBP, LBP-TOP, DoG are used to capture the facial features	A content-based on AlexNet is proposed which consists of 5 convolutional and max-pooling layers, and 3 fully connected layers.	86.4%	-	-
Tariq <i>et al.</i> (2018) [26]	Reenactment + Replace- ment	Facial-based features are extracted	A neural network is proposed using ensem- ble methods to detect GAN generated and human created fake images.	-	-	0.94
Ciftci <i>et al.</i> (2019) [135]	Reenactment + Replace- ment	Spatio- temporal based fea- tures of face	A novel biological sig- nal map is introduced to train the neural network for better au- thentic classification in the wild.	96% for Face- Forensic dataset and 91.07% for own collected dataset.	-	-

Table 2.20 continued from previous page

Li <i>et al.</i> (2018) [136]	Reenactment + Replace- ment	Face warping features are extracted from images	The proposed method detects the artifacts in the images by com- paring the surround- ing regions and gener- ated face areas with a CNN model to distin- guish deepfake videos.	-	-	0.932
Amerini <i>et al.</i> (2019) [137]	Reenactment + Replace- ment	CNN based techniques are applied to learn different features from every frame of sequence.	A new forensic-based technique to identify real and fake video sequences is discussed and applied.	81.6%	-	-

Table 2.20 continued from previous page

Korshunov <i>et al.</i> (2018) [138]	Reenactment	Lip moment, body pose, mouth region, hand detection, face landmarks and 13 MFCC audio based features are used.	An automated system is developed to detect the inconsistencies for audio-visual features using LSTM model.	-	22	-
Korshunov <i>et al.</i> (2019) [139]	Replacement	Visual features for mouth, facial landmarks, mouth movements based features. For audio features, MFCC and senote-based DNN embeddings are used.	Several detection approaches are analysed using recurrent network and phonetic features. For automatic speech recognition, MFCC features are replaced by DNN embeddings to achieve significant performance improvement.	-	16.4	-

Table 2.20 continued from previous page

Chan <i>et al.</i> (2019) [140]	Replacement	Frame-based features are retrieved	A reliable face synthetic tool is developed to distinguish synthesized face videos from real ones by applying pose-to-appearance mapping to generate the target results.	97%	-	-
Yu <i>et al.</i> (2019) [141]	Reenactment	Fingerprints-based features from images are extracted	To classify an image as fake or real, GAN-fingerprint based learning is performed towards image features.	99.6%	0.53	
Wang <i>et al.</i> (2019) [142]	Reenactment	Layer-wise neuron behavior is chosen to retrieve facial features	A neuron coverage based fake detection approach is to detect the monitoring of face recognition system layer by layer spotting the AI fakes.	85%	-	-

Table 2.20 continued from previous page

Li <i>et al.</i> (2019) [143]	Reenactment	X-Ray based facial features	A novel face forgery detection based on face X-ray obser- vations has been proposed by the authors.	-	-	0.992
-------------------------------------	-------------	--------------------------------	--	---	---	-------

To address RQ1, year-wise status of research studies have been done. The citation information of every research article have been used to identify the impact of publications for addressing RQ2. Various types of FCD techniques and types of fake content have been explored in this chapter to answer RQ3, RQ4 and RQ5. Limitations of open-source tools available are also discussed to address RQ6.

2.9 Discussions

It can be concluded from the comparative analysis of related work done in the field of fake articles that the research work done in the field of fake content detection is mainly restricted to SVM and Naïve Bayes classifiers using only N-gram and Tf-Idf features extraction approaches. Limited work has been done on MLP, LSTM models and hashing based extraction approach which also accounts for better efficiency. Also, the existing fake content detection models are built using supervised machine learning algorithms whereas manual hand-crafted feature extraction is more time consuming and inefficient method to achieve the best accuracy. Various methods discussed in this chapter are both painstaking and time-consuming tasks as they used hand-crafted

features for detecting fake short textual content and its classification. Less work has been done to automatically identify the type of clickbait headlines on the basis of short text. For deepfake content, it has been analysed that the research work done in the field of deepfake video or image detection is mainly restricted to CNN and SVM classifiers. Limited dataset is available for face-swapped deepfake videos of famous politicians. These existing techniques studied so far provides a direction to be followed further for quantitative and qualitative research.

Chapter 3

Implementation of Fake News Article Detection System

A growing interest related to fake content detection has attracted many researchers as fake information is circulated through online social media platforms such as Facebook, Twitter [72, 12, 13], Instagram [77], WhatsApp [15, 9], *etc.* The fake content is spreading at a faster pace to gain popularity over social media, to distract people from the current critical issues. Most of the people believe that the information they receive from various social media sites is reliable and true, *i.e.*, people are inherently truth-biased. Also, people easily trust and want to believe in what they actually interpret in their minds, *i.e.*, confirmation-biased. In general, it has been analysed that humans are unable to recognize deception effectively. Due to which a serious and negative impact of fake articles can be seen on society and individuals leading to an imbalance of the news ecosystem. It was observed that during US president election [65], most of the widely spread articles on social media were fake. Recently a fake video related to Kerala battling with

floods was viral on social media platform (Facebook). Also, during India's national election (2019), various WhatsApp groups ($> 900,000$) were created to disseminate the fake information regarding India's ruling party [144].

There are several reasons for the change in behavior for such kind of consumptions. The content shared on social media platforms requires less time and cost than on newspapers or traditional news media. Easier to share content in the form of video, blogs, posts with friends or users. The chapter contributions are discussed in next section.

3.1 Contributions

Researchers have analysed that an automated system sometimes identifies fake news articles better than human do. The automated systems can play an important tool for identifying fake stories, articles, blogs, clicks which manipulate public opinion on social networking sites [75, 145]. Taking into need for the development of such fake news detection system, we have identified news articles as fake or real by using supervised machine learning classifiers such as Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM), Linear models, Neural Networks (NN) and Ensemble models. To get effective results, three different datasets (News Trends, Kaggle and Reuters) have been collected with similar characteristics.

The feature extraction techniques, Term Frequency-Inverse Document Frequency (Tf-Idf), Count-Vectorizer (CV), Hashing-Vectorizer (HV) are used to extract feature vectors from the textual content of articles.

The effectiveness of a set of classifiers is observed when they predict the labels of the testing samples after learning the training data using these features extraction techniques.

Various supervised ML models are extensively used for categorization of textual data as fake or real, but such models were not able to obtain the results of an ideal classifier. So, a multi-level voting model has been proposed to build an ideal classifier with significant improvement than previously existing models. The main contributions are as follows.

- Statistical analysis of collected datasets (News Trends, Kaggle and Reuters) with negative and positive instances has been performed.
- Twelve existing ML models are evaluated using Tf-Idf, CV, HV feature extraction techniques to retrieve the best model based on performance metrics.
- A novel method is proposed by merging the existing ML models based on false prediction rate.
- Proposed an ideal multi-level voting classifier (three-level) to verify the effectiveness of the ensemble model.
- A comparative study is performed to show the effectiveness of our proposed model.

The performance of the multi-level voting system is analysed using parameters like precision, recall, specificity, ROC curve, F1-score.

3.2 Problem Statement

To address the issue of fake content generation and dissemination through various online social platforms, an appropriate feature extraction technique is chosen to improve the efficiency of the existing ML classifiers. A novel

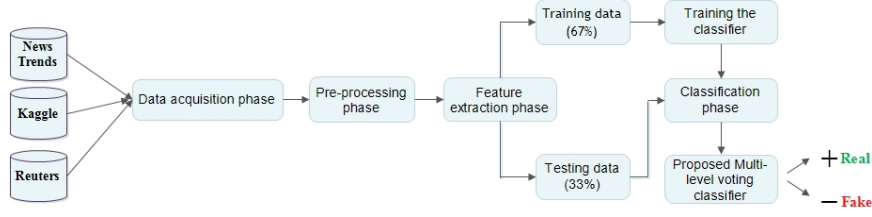


Figure (3.1) Process of the proposed automatic fake content detection system

multi-level voting ensemble model has been proposed to develop an efficient fake news article detection system. Mathematically, the problem statement can be represented as- To identify $S = \{fake, real\}$ for a document D where $D = \{t_1, t_2, \dots, t_n\}$ and t_i represents the text in a news article a_i chosen from corpus with series of engagements that is composed of title, body and label of the article as $e_{ijk} = (t_i, b_j, l_k)$. The task is to evaluate and analyze the best feature extraction method f_m where $m = \{Tf-Idf, CV, HV\}$ using machine learning classifier to compute high efficiency in our proposed system. The approach followed in this chapter has been discussed in the next section.

3.3 Methodology

The generic process of the proposed fake news article detection system is seen in Figure 3.1. To train the system three corpora has been collected from three different sources by downloading the datasets from News Trends, Kaggle and Reuters websites. In the pre-processing phase, stop words and duplicate text from news articles is removed. The missing values, *i.e.*, not available (NA) values [64] are collected and cleaned in the next step. The data retrieved is then split into two parts, training (0.67) and testing (0.33) sets. The feature extraction phase is then carried out to retrieve meaningful

features from the textual data. In this phase, the features are extracted from the articles. Three feature extraction techniques such as Term Frequency-Inverse Document Frequency (assigns weights according to the importance of the terms in the document), Count-Vectorizer (counts the frequency of the terms in a document) and Hashing-Vectorizer (follows the hashing trick) have been applied. The features retrieved are then fed to the classification algorithm chosen in next phase. The various ML models such as MultinomialNB, Passive Aggressive, Stochastic Gradient Descent, Logistic Regression, Support Vector Classifier, Nu-Support Vector Classifier, Multi-Layer Perceptron, Linear-Support Vector Classifier, AdaBoost, Gradient Boosting, Decision Tree, Voting classifiers [146] are chosen to learn and identify the patterns and outcomes from them. The models are then evaluated based on performance metrics to achieve an efficient classifier. Based on the analysis done, the models are integrated to propose a multi-level voting model to achieve high efficiency and then is compared with the existing work [62]. The detailed working of each phase, that has been implemented in Python framework is discussed below.

3.3.1 Data Collection

There are many sources of fake article generation such as Facebook [74], Twitter [68, 147, 73, 63], *etc.*, which are used as a trading platform to disseminate fake contents. We have used News Trends [23], Kaggle [24] and Reuters [25] dataset with similar attributes such as headlines, body, publisher name of the article, published date, categorical and missing values. The News Trends, Kaggle, and Reuters corpus consist of 7,795; 4,048 and 21578 news articles labeled as fake and real news, respectively. Overall statistics of our three collected datasets are discussed in Table 3.1. To analyse the

Table (3.1) Statistics of collected corpus

Corpus	Total article	Cleaned articles	Real articles	Fake articles	Published year
News Trends	7,795	6,335	3,171	3,164	2017
Kaggle	4,048	3,983	1,865	2,118	2017
Reuters	21,578	19,969	9,622	10,347	2004

length distribution of the titles for fake and real articles, Figure 3.2 is visualized. Where the X-axis labeled as ‘title’ represents the number of terms used in news article title’s/headlines whereas Y-axis represents the corresponding number of articles having the same length distribution. A conclusion can be drawn from the mean distribution by visualising Figure 3.2 that the length of the title’s or headlines of fake news articles is often longer than real news articles. To further analyse the length distribution of the body content in articles, Figure 3.3 is observed, where the X-axis labeled as ‘text’ represents the number of terms used in news article texts whereas Y-axis represents the corresponding number of articles having the same length distribution.

It has been analysed that the length of body/text of real news articles are often longer than fake news articles as shown in Figure 3.3a and Figure 3.3b but for Reuters corpus the length of real news articles is more than fake news articles as seen in Figure 3.3c.

The statistics of the mean distribution for Figure 3.2 and Figure 3.3 are compared in Table 3.2. In general, a conclusion can be drawn after analysing different datasets that the headlines of fake articles are longer in length and have shorter body content than real articles published over social networking sites [80].

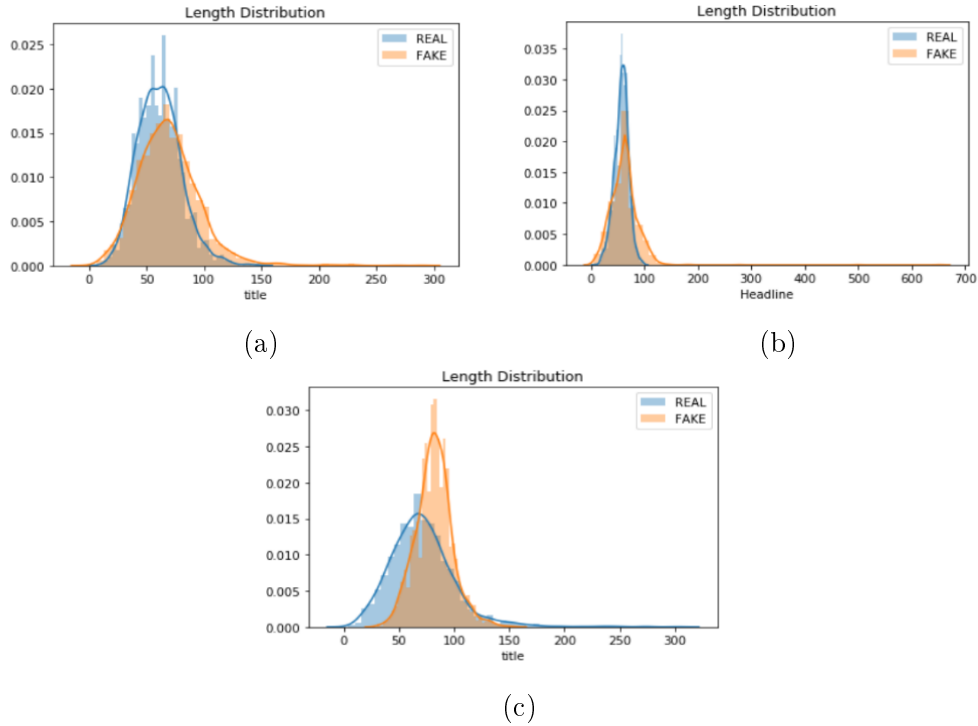


Figure (3.2) Length distribution of headline for fake and real news articles on (a) News Trends (b) Kaggle, and (c) Reuters datasets

3.3.2 Pre-processing

In the pre-processing phase, non-semantic words such as prepositions, conjunctions, pronouns, *etc.*, also known as stop words are removed from the textual document as they provide very little or no information about fake content of an article. The redundant data in form of textual strings is removed from the document using a regular expression (*regex*) in the next step as shown in Figure 3.4.

The *regex* and *pandas* library has been used to perform the pre-processing task [16]. *Regex* library has been used in Python to define a search pattern using a sequence of characters whereas *dropna* method from *pandas* is used

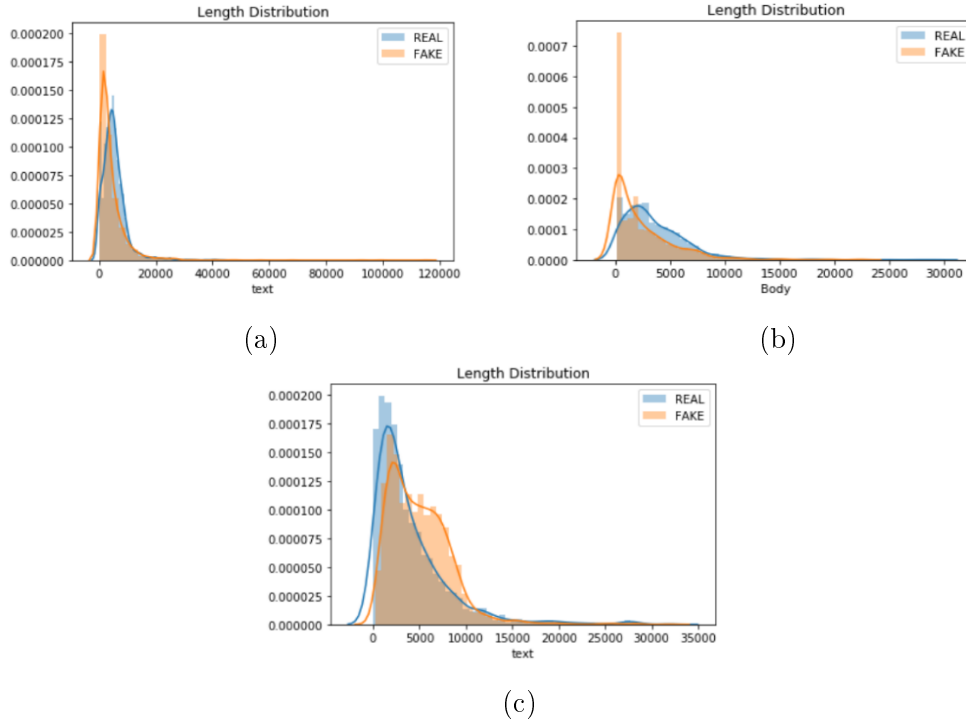


Figure (3.3) Length distribution of text for fake and real articles on (a) News Trends (b) Kaggle, and (c) Reuters datasets

for cleaning the missing values in Python dataframe. We have used *random_state* function to select the entries from the dataset which is further used to split training and testing data points as it is used to split the data randomly.

To avoid over-fitting, three standard splits (70:30, 67:33, 60:40) were used to perform the experiment. When the first standard split (70:30) was performed, it was observed that the data point dealt with an issue of over-fitting. During the second split (60:40), under-fitting of data was analysed. Whereas, the third split (67:33) gave the best predicted line covering the majority of data points in the graph, so a standard split of 67:33 was chosen. The train-

Table (3.2) Mean distribution of labeled articles for News Trends, Kaggle and Reuters corpus

Corpus	Mean distribution of title labeled as fake	Mean distribution of title labeled as real	Mean distribution of text labeled as fake	Mean distribution of text labeled as real
News Trends	69.18	61.38	4121.04	5292.16
Kaggle	62.47	57.32	2380.82	3544.84
Reuters	81.79	71.29	5156.08	4540.26

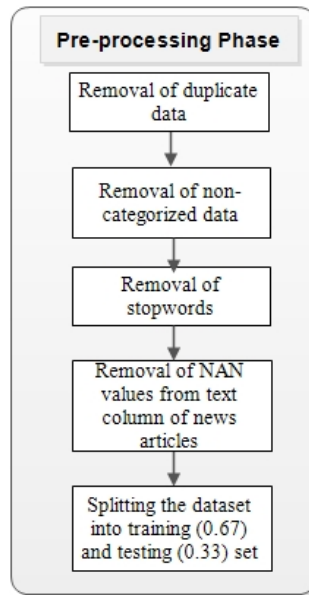


Figure (3.4) Steps performed during the pre-processing phase

ing data is then fed to feature generation phase as discussed in the next section.

3.3.3 Feature Generation

In order to extract numerical features from a textual document, tokenization, counting and normalization is done. During tokenization, each word is given a unique integer *ID*, following which occurrence of tokens is counted and then

normalization of such tokens takes place. The whole process of converting the textual document into numerical feature vector is called as vectorization. Together this strategy (tokenization, counting, normalization) is called as ‘Bag of n-grams’ or ‘Bag of Words’ where n represents the continuous sequence of terms [148]. The three feature extraction techniques as shown in Figure 3.5 for retrieving features from the textual content of an article are Count-Vectorizer, Tf-Idf and Hashing-Vectorizer which use *CountVectorizer*, *TfidfVectorizer* and *HashingVectorizer* classes from *feature_extraction* library of Python, respectively.

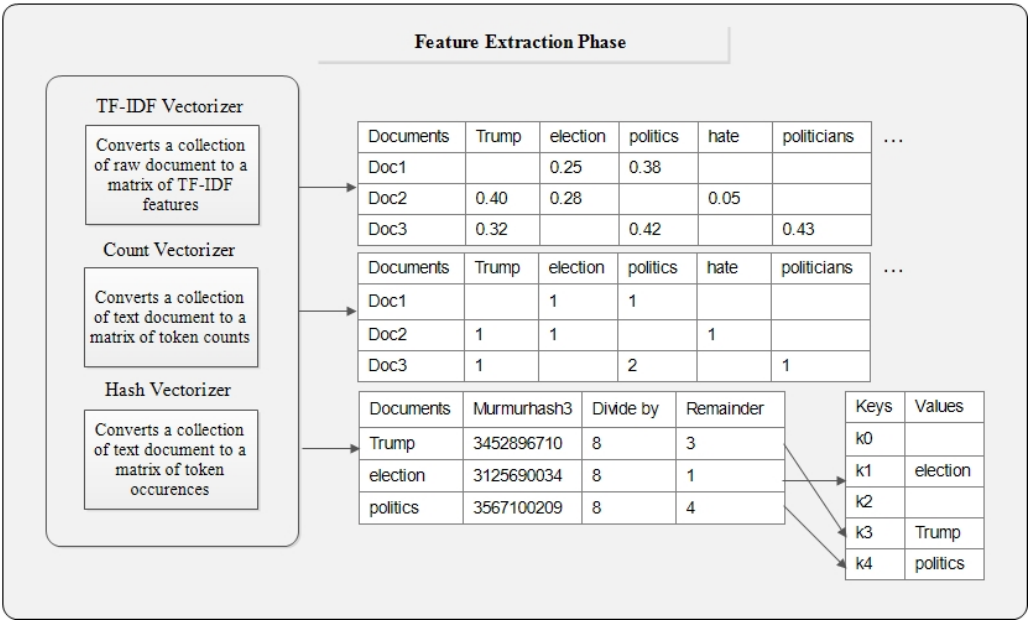


Figure (3.5) Conversion of textual content into numerical vectors through Tf-Idf, Count-Vectorizer and Hashing-Vectorizer feature extraction techniques

Count-Vectorizer (CV)

CV is absolutely based on count of the word occurrences in the document. In this technique, both counting the occurrences of tokens and tokenization process is performed. It has many parameters for refining the type of features. One can build features using any of the three parameters, unigram ($min_df=1$), bigram ($min_df=2$) and trigrams ($min_df=3$). We have used min_df as 1 for our experiment. Here, each vector (term) in a document represents the individual feature name and its occurrence is depicted through a matrix to make it easier to understand as shown in Table 3.3. It has been

Table (3.3) Sparse matrix representation using a Count-Vectorizer feature extraction technique

Document	Narendra	elections	vote	politics	Punjab	BJP	candidate
Doc1	0	1	2	0	0	2	1
Doc2	4	0	1	0	0	1	0
Doc3	1	3	2	0	1	0	1

observed from the above table, after pre-processing phase, the terms retrieved from the documents are represented as vectors on top of the sparse matrix and the frequency of terms in particular document is represented through count occurrences. Tag clouds of top 30 features retrieved after executing CV method is shown in Figure 3.6. It was observed that words like *corruption*, *attacking*, *islamic*, *obama*, *losing*, *com* are seen under fake tag clouds.

CV also counts the number of words occurring more frequently in the document, which may overshadow the words which occur less frequently but may have more importance to the document feature. This limitation of CV can be handled by using Tf-Idf feature extraction technique as explained below.

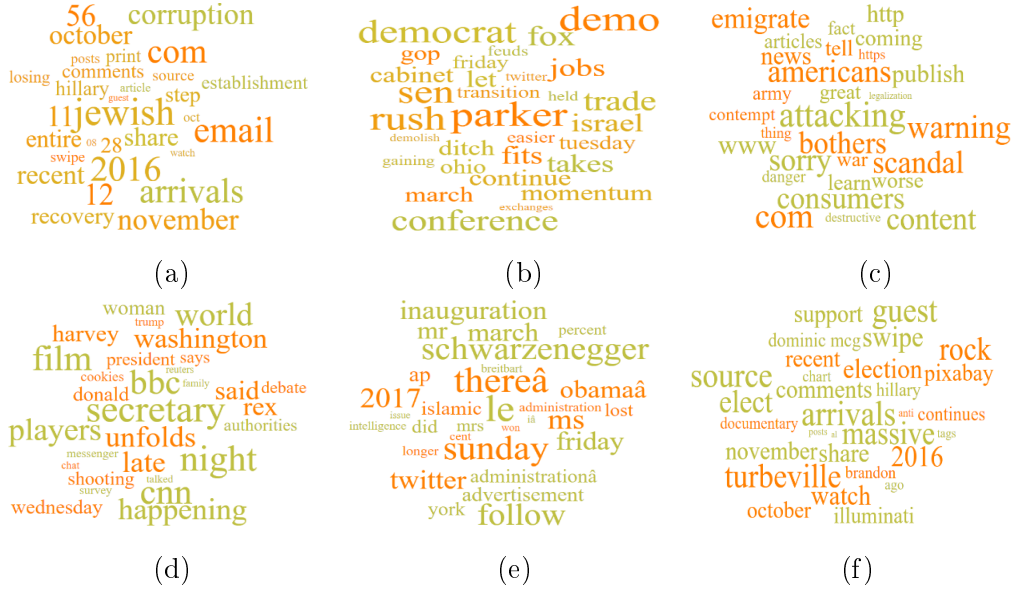


Figure (3.6) Top 30 (a) News Trends fake, (b) News Trends real, (c) Kaggle fake, (d) Kaggle real, (e) Reuters fake, (f) Reuters real word clouds using count-vectorizer feature extraction technique

Term frequency- Inverse Document frequency (Tf-Idf)

Tf-Idf is a weighing matrix which is used to measure the importance of a term (count + weight) to a document in a dataset. The tokens retrieved from the textual data using both Tf-Idf and CV techniques are same, but weights assigned to the tokens of both techniques are different. Tf-Idf is composed of two metrics, named as, term frequency (tf) and inverse document frequency (idf). Tf-Idf is represented by Eq. (3.1).

$$tfidf = tf(t, d) \times idf(t, d) \quad (3.1)$$

Here, Term Frequency denoted as tf and is calculated from the count (c), term (t) in document (d) and represented as $tf(t, d) = c_{td}$. The frequency of occurrence of words to a binary feature is converted by using 1 (present

in document) and 0 (not present in document). The frequencies can be normalized using average and logarithms. The inverse document frequency (*idf*) for a word w in document text (t) as computed by Eq. (3.2).

$$idf(t, d) = 1 + \log \frac{T}{(1 + df(t))} \quad (3.2)$$

Here, T represents the total document count in our corpus and $df(t)$ represents the count of the number of documents where the term t is present. The product of two measures will help to compute *tfidf*. Euclidean's normalized form is used to calculate the final Tf-Idf metric as given by Eq. (3.3).

$$tfidf = \frac{tfidf}{||tfidf||} \quad (3.3)$$

Here $||tfidf||$ is the Euclidean norm. Tag clouds of top 30 features for fake and real articles using Tf-Idf feature extracting technique is shown in Figure 3.7. It was observed that words like *www*, *danger*, *sorry*, *wars*, *islamic* are most common under fake news articles. The difference between CV and Tf-Idf approach is that the tokens retrieved from textual data are same but both have different weights assigned to each token being extracted.

Hashing-Vectorizer (HV)

It is a memory-efficient technique. Unlike previous two techniques, tokens are stored as a string and here hashing trick is applied to encode the features as numerical indexes. Let us discuss the concept of HV using an example shown in Figure 3.8. When data is entered, the hashed attributes of data are retrieved. The hashed terms like *Trump*, *election* and *politics* are extracted from the document. In the next step, the hashing trick is applied on the hashed attributes, where a *Murmurhash3* function is applied on the hashed terms to generate a random number. Further, the assigned random numbers

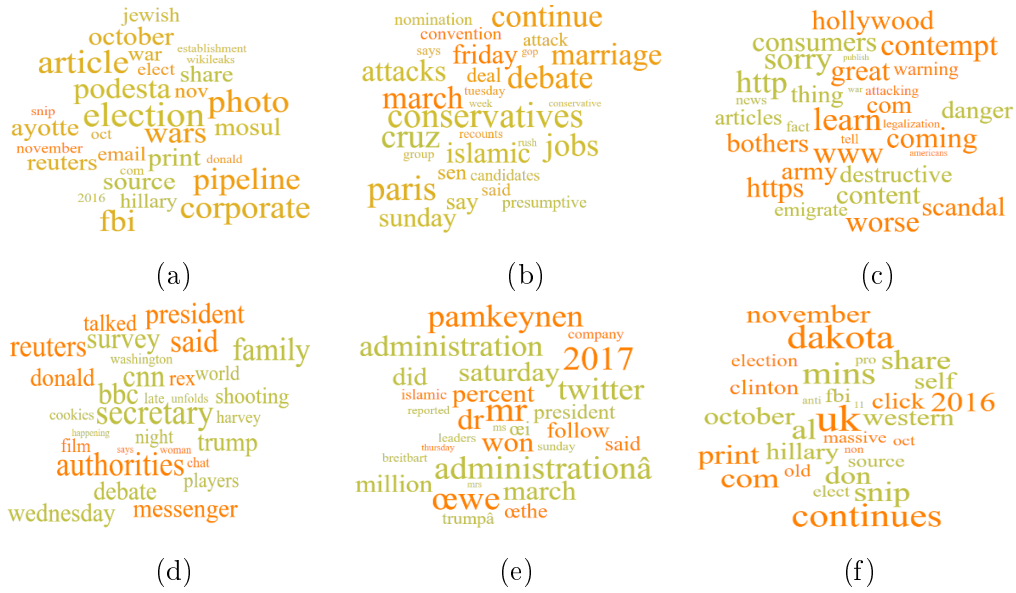


Figure (3.7) Top 30 (a) News Trends fake, (b) News Trends real, (c) Kaggle fake, (d) Kaggle real, (e) Reuters fake, (f) Reuters real word clouds using Tf-Idf feature extraction technique

are divided by 8 and are stored in different keys such as k_2 , k_3 , k_4 based on the remainders retrieved after applying the *murmurhash3* function which is used for hash-based lookup.

There is a possibility of collision when data have equal hashed attributes. Let us suppose in our example document, we have *Trump* and *politics* words as an important keys which are seen more than once, thus causing collisions at k_3 , k_4 positions. The collided values are then occupied by other vacant positions in a set of documents. Such collision processing is dealt with parallel processing. The process is conceptually explained in Figure 3.9.

These six terms are assigned six keys as shown in Figure 3.9 and are entered in the hash table. The hash values of keys k_1 , k_3 , k_6 are same, *i.e.*, *Trump*; k_4 and k_5 are same *i.e.*, *politics*, but rest have different values.

Documents	Murmurhash3	Divide by	Remainder	Keys	Values
Trump	3452896710	8	3	k0	-
election	3125690034	8	2	k1	-
politics	3567100209	8	4	k2	election
				k3	Trump
				k4	politics

Figure (3.8) Hashing trick on textual data using Murmurhash3 function to get values in a specific range

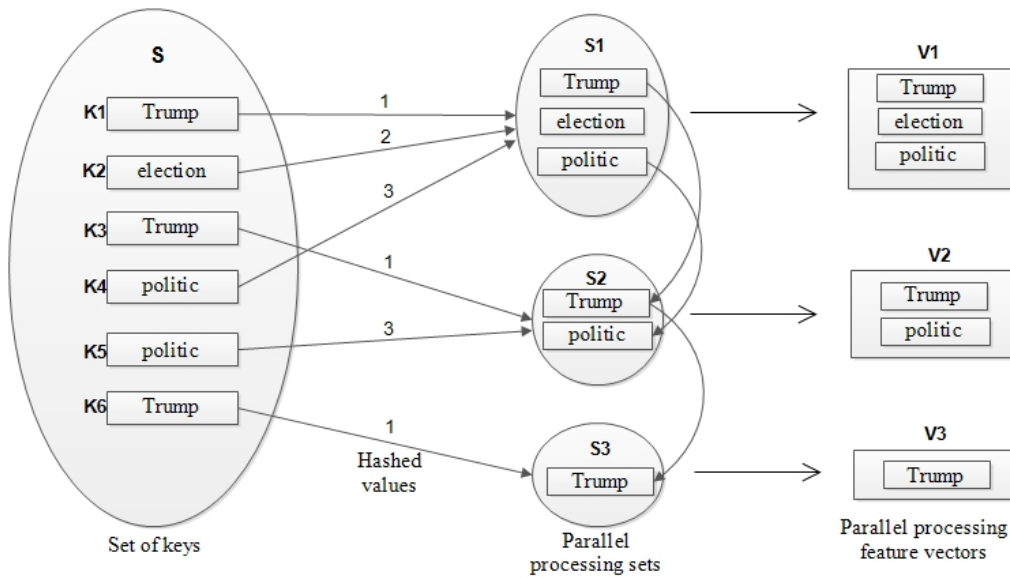


Figure (3.9) Entry of redundant data into a hash table during parallel processing

Due to collision $k1$, $k3$ and $k6$ cannot be placed in same set ($S1$). To enable parallel processing, $k1$, $k3$ and $k6$ are placed in different sets. No two same hash values can be placed in a single set. Different keys like $k1$, $k2$ and $k4$ can be placed in the same set ($S1$) as these keys have different values. Values at keys, $k3$ and $k6$ are same so cannot be processed parallelly, therefore are processed in different sets. The values in $S1$, $S2$, $S3$ are organized into vectors (numerical features) and can be processed using vector operations.

The drawback of HV is that there is no way to get the feature names from feature indices, *i.e.*, the inverse transform cannot be used to compute the most important feature names through HV, unlike rest two methods.

All three methods are used for converting text data into vectors. The motivation of using CV approach is to get an entire vocabulary and an integer count for the number of times each word appears in the document. The CV method includes most frequent words like ‘the, in, on’, which does not signifies any importance in the document. So, Tf-Idf is used to offer a downscale of a given words appearance in the document. To reduce the vocabulary size taken in previous two methods, HV is used. Here, no vocabulary is required and an arbitrary fixed length can be chosen by the user. Experiments are performed using all these three approaches.

3.4 Classification Algorithms

The processed dataset retrieved after pre-processing and feature extraction phase is then fed to the classification phase for the identification of fake news article. Six machine learning techniques, *i.e.*, Naïve Bayes (MultinomialNB), Support Vector Machine (Support Vector Classifier (SVC), NuSVC, LinearSVC), Decision Tree (CART), Linear (Passive Aggressive (PA), Stochastic

Gradient Descent (SGD), Logistic Regression (LR)), Neural Network (Multi-Layer Perceptron (MLP)) and Ensemble models (AdaBoost, Gradient Boosting, Voting) have been applied as shown in Figure 3.10.

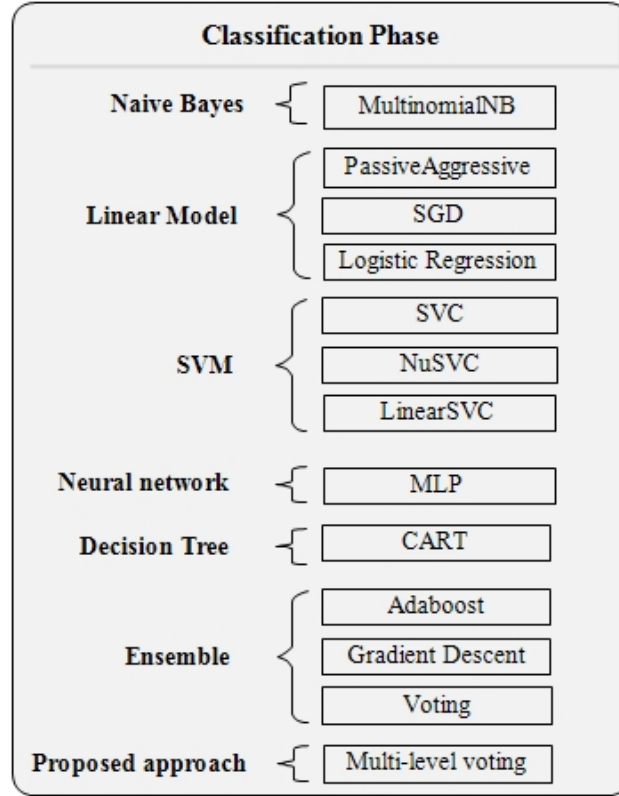


Figure (3.10) Classification phase

Classifier functions help to map the input feature vectors $f \in F$ to output labels $l \in \{1, 2, 3, \dots, n\}$, where F is the feature space. The feature space is represented as, $F = \{Fake, Real\}^R$, where R is the real number. Our aim is to learn the classifier function from a labeled training data.

3.4.1 Naïve Bayes (NB)

It is a type of probability classifier. It works on Bayes Theorem and handles both categorical and continuous variables. NB assumes that every pair of features with labeled value is independent of each other. Given a collection of D documents from news articles, $D_i = \{d_1, d_2, \dots, d_n\}$, where each document consists of T terms such as $D_i = \{t_1, t_2, \dots, t_m\}$. Then the probability of D_i occurrence in class label C_l is given by Eq. (3.4).

$$P(C_l|D_i) = P(C_l) \prod_{n=1}^m P(d_n|C_l) \quad (3.4)$$

Here, the conditional probability of term t_m present in a document of class label C_l and the prior probability of document occurring in class label C_l is denoted by $P(C_l)$.

MultinomialNB is a type of Naïve Bayes algorithm used for text classification. Data used in text classification to apply MultinomialNB can be represented as *Tf-Idf* vectors, hashing vectors and count vectors. The feature vectors $V_f = (V_{f1}, V_{f2}, \dots, V_{fn})$ are parameterized for each class C_n in the distribution, where n represents the feature numbers. The likelihood of observing V_f is given by Eq. (3.5).

$$P(C_n|V_f) = \frac{X_i!}{\prod_{n=1}^p} f_n! \prod_{n=1}^p V_n f_n \quad (3.5)$$

Here, f_n is the number of times the n^{th} feature has occurred in the document, X_i is the number of draws taken from the bag of features. $V_n^{f_n}$ and $f_n!$ are computed from training data.

3.4.2 Linear Model

The linear model helps to classify the group by making linear combinations of feature vectors. Linear classifiers work well with many features but it works best for document (features are extracted from the text) classification. If $\vec{v}$ is the input feature vector to the classifier, then the resultant score is given by Eq. (3.6).

$$s = f(\vec{v}\vec{w}) = f\left(\sum_i w_i v_i\right) \quad (3.6)$$

Here $\vec{w}$ is the weight of a feature vector and the function f gives the desired output of two vectors. The three linear models used are Passive Aggressive (PA), Stochastic Gradient Descent (SGD) and Logistic Regression (LR) classifiers. The PA algorithm has similar behavior with perceptron classifier in terms of learning rate but has dissimilar behaviour in terms of regularization parameter. The PA classifier is equivalent to PA-l [74] when the loss parameter is *hinge* and PA-ll [74] when the loss parameter *squared_hinge*. Second linear model used is SGD. The SGD model is updated with the decreasing learning rate after each sample interval and by default the loss parameter used is *hinge*. The classifier also allows minibatch learning. The LR model can be used for both multi and binary problem classification. Any other input format apart from *float64* gets converted in this classifier. All three linear models discussed above can take both sparse and dense matrix as their input.

3.4.3 Support Vector Machine (SVM)

SVM works on Structural Risk Minimization Principle. It is defined by a ‘best’ separating hyperplane and is also called as a discriminative classifier. Through the SVM model, feature vectors retrieved from text document of

news articles are represented as points in feature space. Then the feature vectors are mapped in such a way that a wide gap is visible to perform linear classification. In our collected dataset the feature vectors are marked by making two categories, $C = \{Fake, Real\}$, then the training classifier builds a model which assigns new feature vectors to both defined categories.

SVM classes such as Linear Support Vector Classifier (LinearSVC), Nu-Support Vector Classifier (NuSVC), Support Vector Classifier (SVC) are used for performing classification on the dataset. NuSVC and SVC are almost similar but use slightly different sets of parameters and their mathematical formulations also vary. Whereas LinearSVC is another type of Support Vector Classification (SVC) and uses the case of linear kernel. All three classes take input in the form of two arrays with array X having 2-dimensional size [sample_number, feature_vectors] to handle training data and array Y having 2-dimensional size [category_label, sample_number]. The Decision function is the same for both SVC and NuSVC is given by Eq. (3.7).

$$sgn(\sum_{f=1}^n y_f \alpha_f K(V_f, V) + \mu) \quad (3.7)$$

Where V_f are the training feature vectors, $f = 1, 2, \dots, n$ in two categories. $K(V_f, V)$ is the kernel and $y_f \alpha_f$ is the dual coefficient parameter which holds support vectors and an independent intercept term μ . The only difference between SVC ($C=[0, \infty]$) and NuSVC ($C=[0, 1]$) is seen from parameter C which is the penalty parameter of the error term. The class LinearSVC supports both sparse and dense input and is implemented in terms of liblinear so is more flexible in terms of the loss function and penalties, and scales better to large testing samples.

3.4.4 Neural Network (NN)

Neural networks are composed of highly interconnected neurons to solve specific problem parallelly. Multi-Layer Perceptron (MLP) is implemented on our collected dataset. The classifier can be trained on either regression or classification dataset. The feature vectors $V_f = \{v_1, v_2, \dots, v_n\}$ are retrieved after feature extraction phase and through training dataset the classifier learns a given function in Eq. (3.8).

$$f(n) : Ri \rightarrow Ro \quad (3.8)$$

Where i is the dimensions for input and o is the dimensions for an output. In MLP there can be one or more non-linear layer (hidden layer) between the input and output layer. The input layer made up of neurons, where each neuron represents the input feature which is fed into the hidden layer. Then the hidden layer computes the weighted summation $w_1v_1 + w_2v_2 + \dots + w_iv_i$, followed by function $f(n)$. The output value is given by the last hidden layer and is received by the output layer.

3.4.5 Decision Tree (DT)

The DT classifiers can be used for both regression and classification. The classifier predicts the target variable by learning the feature data and dividing the area into sub-regions. Based on two criteria multiple features are divided, one is a measure of impurity and other is information gain. In our dataset ‘*gini*’ is the chosen impurity measure to calculate the highest information gain at each node for dividing the DT. In case of document data, the conditions depend upon the particular term in a text document of the news article. The data is divided repetitively until the leaf node cannot be further

divided acquiring least information on them. The majority count of labels in the leaf node are used for classifying the textual data.

3.4.6 Ensemble methods

Such methods help to build a learning algorithm by combining the estimators to get robust classifier over single classifier. The boosting methods implemented on our dataset are Gradient Descent Boosting (GDB) and AdaBoost classifier for binary classification. The AdaBoost classifier assigns more weight to feature vectors which is difficult to handle and less weight to the features which can be easily handled. This process is repeated until the classifier correctly classifies the training data. The GDB model works on three elements such as weak learner, loss function and additive model as discussed in [7].

The other type of classifier that can be useful for balancing individual weakness of ML models is Voting classifier. The Voting classifier has predicted the categories based on ‘hard’ voting which classifies the sample based on majority class label. To evaluate the classifiers discussed above, the performance metrics are defined and corresponding experimental results are discussed in the next section.

3.5 Evaluation Phase

The performance measures for binary classifiers applied has been evaluated with the help of a confusion matrix defined by four cells as shown in Table 3.4, where:

- cell ‘a’ counts the predicted document as ‘Fake’ when actually it is

‘Fake’, known as true positive (TP) rate.

- cell ‘b’ counts the predicted document as ‘Real’ when actually it is ‘Fake’, known as false positive (FP) rate.
- cell ‘c’ counts the predicted document as ‘Fake’ when actually it is ‘Real’, known as false negative (FN) rate.
- cell ‘d’ counts the predicted document as ‘Real’ when actually it is ‘Real’, known as true negative (TN) rate.

Table (3.4) A confusion matrix representation

Actual↓ Predicted→	Fake news article	Real news article
Fake news article	TP (a)	FP (b)
Real news article	FN (c)	TN (d)

The conventional performance measure has been evaluated from the above confusion matrix cells. The measures computed from the matrix are precision as represented by Eq. (3.9), recall by Eq. (3.10), specificity by Eq. (3.11), accuracy by Eq. (3.12), error by Eq. (3.13), F1-score by Eq. (3.14) as shown below.

$$Precision (Pr) = \frac{a}{a + b} \quad (3.9)$$

$$Recall (Re) = \frac{a}{a + c} \quad (3.10)$$

$$Specificity (Sp) = \frac{d}{d + b} \quad (3.11)$$

$$Accuracy (Acc) = \frac{a + d}{n}, \text{ where } n = a + b + c + d > 0 \quad (3.12)$$

$$Error (Err) = \frac{b + c}{n}, \text{ where } n = a + b + c + d > 0 \quad (3.13)$$

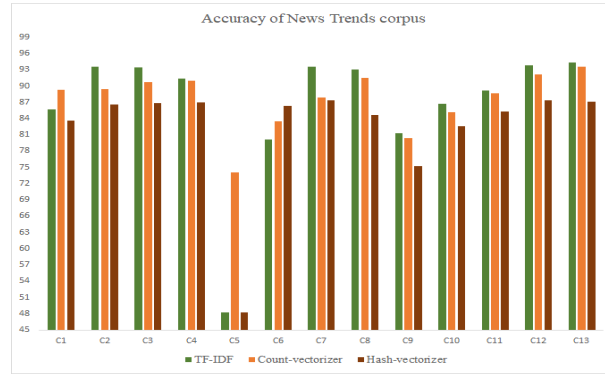
Table (3.5) Comparative analysis of accuracy measure using machine learning classifiers

MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
Multinomial Naïve Bayes	85.7	89.3	83.6	93.2	95.4	89	84.8	89.9	86.2
Passive Aggressive	93.5	89.4	86.6	98.3	97.6	95.7	96.2	94.3	88
Stochastic Gradient Descent	93.4	90.7	86.8	98	97.1	95.5	96.2	94.8	89.4
Logistic Regression	91.4	91.0	87.0	96.4	98	93.6	94.9	95.7	89.6
Support Vector Classifier	48.2	74.1	48.2	52.9	74	52.9	52.5	74.3	81.3
NuSVC	80.1	83.5	86.3	92.2	86.1	92.6	70	86.6	88.3
LinearSVC	93.6	87.9	87.3	97.9	97.6	95.3	96.3	94.4	89.3
Multi-Layer Perceptron	93	91.5	84.7	97.1	97.3	94.7	96.2	96.2	90.1
Decision Tree	81.3	80.4	75.2	95.6	96.2	91.3	87.7	88.5	80.6
AdaBoost	86.7	85.1	82.6	97.3	96.6	92.9	92.5	92.4	85.9
Gradient Boosting	89.2	88.6	85.3	98.5	98.0	95.7	92.3	92.5	87.6
Voting Classifier	93.8	92.1	87.3	98.3	98.3	95.9	96.1	96.4	90.3

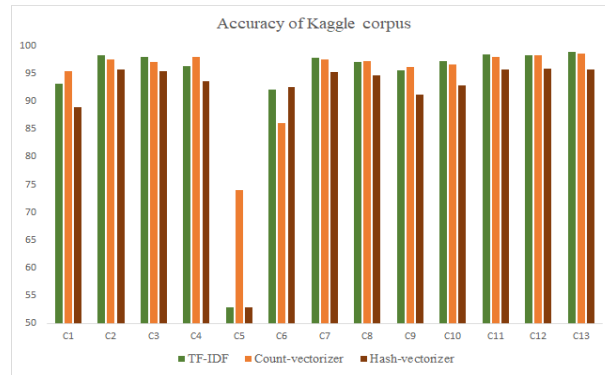
$$F1 - score (F1) = \frac{2 \times Pr \times Re}{Re + Pr} \quad (3.14)$$

The predictions made by the classification models are evaluated in this phase based on their performance metrics. The most intuitive performance measure is accuracy, which helps to predict the best model. Various machine learning models used in experiment are MultinomialNB (*C1*), Passive Aggressive (*C2*), Stochastic Gradient Descent (*C3*), Logistic Regression (*C4*), SVC (*C5*), NuSVC (*C6*), LinearSVC (*C7*), Multi-Layer Perceptron (*C8*), Decision Tree (*C9*), AdaBoost (*C10*), Gradient Descent (*C11*), Voting (*C12*) and Multi-level Voting (*C13*) classifiers. To evaluate these models, a comparative analysis has been shown in Figure 3.11.

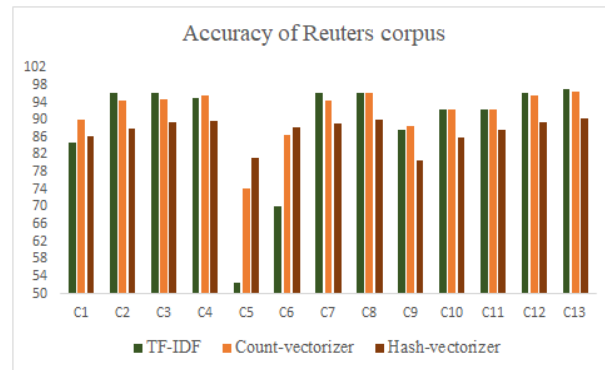
The experiment is performed on three (News Trends, Kaggle, Reuters) different corpus and Tf-Idf, CV, and HV feature extraction techniques are used to extract the feature vectors from the documents of the chosen corpus. In Table 3.5, the accuracy measure of various ML classifiers is compared.



(a)



(b)



(c)

Figure (3.11) Performance analysis using accuracy metric for (a) News Trends, (b) Kaggle and (c) Reuters dataset on basis of Tf-Idf, HV and CV feature extraction techniques

Table (3.6) Comparative analysis of precision metric using Machine Learning classifiers

MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
Multinomial Naïve Bayes	73.3	85.8	88.4	92.3	95.8	84.4	99.1	97.3	89.3
Passive Aggressive	94.5	90.0	87.6	98.7	96.9	97.1	96.6	93.6	91.1
Stochastic Gradient Descent	94.8	89.4	88.1	98.7	97.8	95.8	96.4	94.2	90.0
Logistic Regression	95.4	94.0	89.8	95.4	97.9	94.2	94.7	95.2	89.1
Support Vector Classifier	100	96.6	100	100	97.2	100	100	53.1	82.0
NuSVC	94.5	96.3	90.4	97.5	97.9	91.6	41.9	77.8	85.7
LinearSVC	96.1	88.8	87.8	98.1	96.5	96.2	93.6	94.1	89.4
Multi-Layer Perceptron	93.6	93.1	84.0	97.2	96.5	96.2	92.1	94.3	89.9
Decision Tree	80.2	81.9	75.0	97.2	96.6	92.8	87.1	88.3	80.1
AdaBoost	89.9	88.7	83.6	97.2	96.6	94.1	92	90.6	84.9
Gradient Boosting	91.9	92.6	89.0	98.4	97.8	95.4	90.7	90.4	89.9
Voting Classifier	95.8	94.2	89.2	98.8	97.8	96.8	96.8	95.0	89.9

The best accuracy retrieved by top 3 models among all three corpus is Linear Support Vector Classifier (LSVC), Passive Aggressive (PA) and Logistic Regression (LR) classifiers.

Other parameters used to evaluate the performance measure of classifiers used are precision, recall and F1-score. The precision metric helps to calculate the proportion of news article that are predicted fake and actually also belongs to the fake news article category. The comparative analysis of precision metric is shown in Table 3.6.

The recall metric helps to calculate the proportion of news article that are predicted fake but actually belongs to both fake and real articles. The comparative analysis of recall metric is shown in Table 3.7.

The specificity metric helps to calculate the proportion of news article that are correctly predicted as a real news article known not to be fake. Specificity is measured as inverse of recall metric. The comparative analysis

Table (3.7) Comparative analysis of recall metric using Machine Learning classifiers

MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
Multinomial Naïve Bayes	95.9	91.5	79.7	94.6	95.4	94	77.9	85.4	85.0
Passive Aggressive	92.2	88.1	85.0	98.0	98.3	94.8	96.0	95.3	86.7
Stochastic Gradient Descent	91.7	90.9	84.9	97.5	96.7	95.6	96.2	95.7	89.7
Logistic Regression	87.7	88	84.2	97.7	98.5	93.7	95.5	96.5	90.7
Support Vector Classifier	48.2	65.7	48.2	52.9	67.4	52.9	52.5	96.3	82.3
NuSVC	72.4	75.9	82.6	88.7	80.2	94.2	99.4	95.8	91.4
LinearSVC	91.1	86.4	86	97.8	98.9	94.9	96.6	95.1	90.1
Multi-Layer Perceptron	91.9	89.9	84.1	97.2	98.2	93.8	96.4	95.8	89.8
Decision Tree	80.7	78.3	73.9	94.5	96.1	90.9	89.2	89.6	82.3
AdaBoost	83.5	81.8	80.9	97.5	96.8	92.6	93.6	94.5	87.8
Gradient Boosting	86.4	85.0	82.0	98.7	98.4	96.3	94.2	95.1	89.9
Voting Classifier	91.6	89.7	85.1	98.0	98.9	95.4	95.0	96.6	89.8

of specificity metric is shown in Table 3.8. Accuracy is only measured as a strong metric when both false negative and false positive values are closer to each other else the metric is not considered as a good performance measure. To convey a balance between recall and precision, F1-score performance metric is selected to retrieve the best model. F1-score metric helps to take both false positives and false negatives into account. The comparative analysis of F1-score metric is shown in Table 3.9.

The objective of ROC is to notice the increase in false positive rates (FPR) with an increase in true positive rates (TPR) with a varying threshold of the classifiers. The performance of class models at various thresholds is shown through graphs in Figure 3.12. The curve drawn in the graph is known as Receiver Operating Characteristic (ROC) curve. The ROC for News trends, Kaggle and Reuters are plotted using two parameters such as FPR and TPR

Table (3.8) Comparative analysis of specificity metric using Machine Learning classifiers

MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
Multinomial Naïve Bayes	79.6	87.5	88	91.6	95.2	84.3	98.7	96.4	87.4
Passive Aggressive	94.8	90.5	88.2	98.5	96.6	96.6	96.2	93.1	89.6
Stochastic Gradient Descent	95	90.4	88.6	98.5	97.5	95.3	96	93.7	88.9
Logistic Regression	95.3	94	89.9	94.9	97.4	93.4	94.2	94.8	88.2
Support Vector Classifier	0	94.4	0	0	95.1	0	0	65.3	80.2
NuSVC	92.9	95.4	90.2	96.9	96.5	90.9	61.3	79.6	85.2
LinearSVC	96.2	89.3	88.5	97.8	96.2	95.7	95.9	93.5	88.4
Multi-Layer Perceptron	93.9	93.4	85.1	96.9	96.1	95.6	94.3	93.2	86.1
Decision Tree	81.7	82.4	76.4	96.8	96.2	91.7	86.1	87.3	78.6
AdaBoost	89.9	88.6	84.2	96.9	96.2	93.2	91.3	90.1	83.9
Gradient Boosting	92	92.5	88.9	98.2	97.5	94.8	90.1	89.9	85.2
Voting Classifier	95.9	94.3	89.5	98.6	97.6	96.3	96.4	94.6	88.8

Table (3.9) Comparative analysis of F1-score metric using Machine Learning classifiers

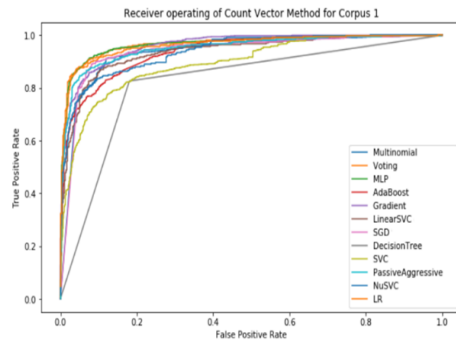
MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
Multinomial Naïve Bayes	86.9	89.4	83.6	93.4	95.5	88.9	87.2	90.9	87.0
Passive Aggressive	93.4	89.2	86.5	98.3	97.5	95.9	96.2	94.4	88.8
Stochastic Gradient Descent	93.3	90.6	86.7	98.0	97.2	95.6	96.2	94.9	89.8
Logistic Regression	91.3	90.9	86.8	96.5	98.1	93.9	95.0	95.8	89.8
Support Vector Classifier	65	77.4	65	69.1	79.6	69.1	68.8	68.4	82.1
NuSVC	81.3	84.5	86.2	92.8	88.1	92.8	58.9	85.8	88.4
LinearSVC	93.5	87.8	87.2	97.9	97.3	94.9	95	94.5	89.7
Multi-Layer Perceptron	92.8	91.6	84.5	97.2	97.3	94.9	93.2	94.1	88.7
Decision Tree	81.1	80.2	75.1	95.8	96.3	91.8	88.1	88.9	81.1
AdaBoost	86.5	85.0	82.5	97.3	96.6	93.3	92.7	92.5	86.3
Gradient Boosting	89.1	88.5	85.3	98.5	98.0	95.8	92.4	92.6	89.9
Voting Classifier	93.7	91.8	87.1	98.3	98.3	96	95.8	95.7	89.8

as given by Eq. (3.15) and Eq. (3.16).

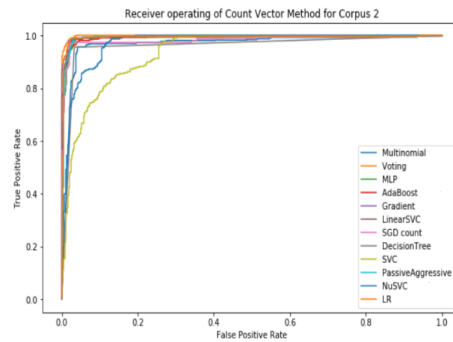
$$TPR = \frac{a}{a + c} \quad (3.15)$$

$$FPR = \frac{b}{b + d} \quad (3.16)$$

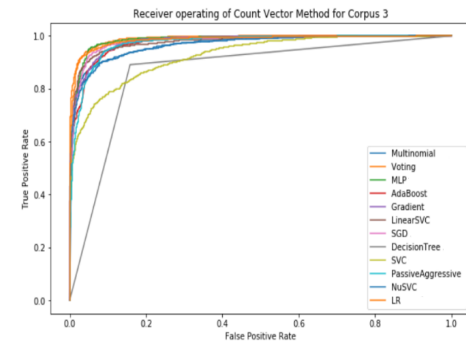
Here, a , b , c , and d represents the TP, FP, FN and TN rates, respectively. predicted document as ‘Real’ when actually it is ‘Fake’, known as false positive (FP) rate, c counts the predicted document as ‘Fake’ when actually it is ‘Real’, known as false negative (FN) rate, d counts the predicted document as ‘Real ’ when actually it is ‘Real’, known as true negative (TN) rate.



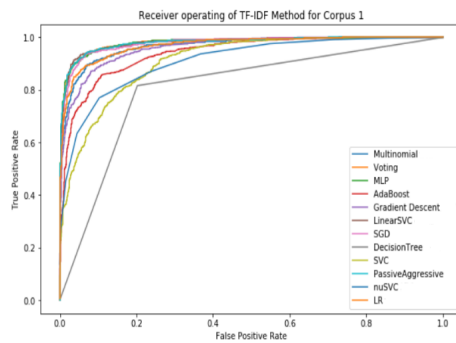
(a)



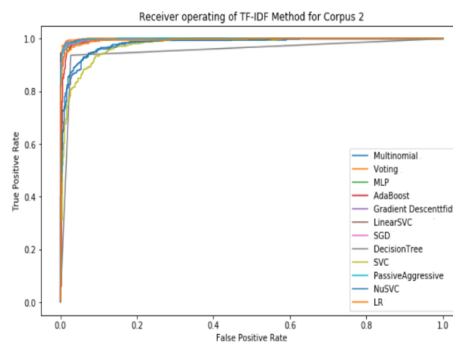
(b)



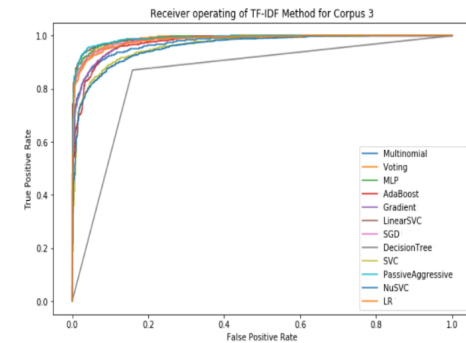
(c)



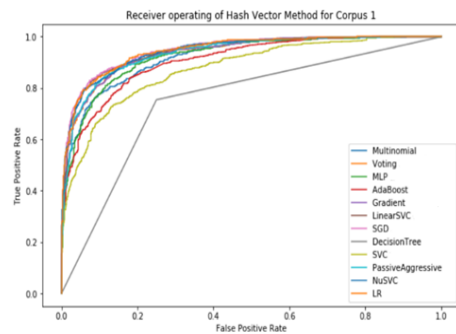
(d)



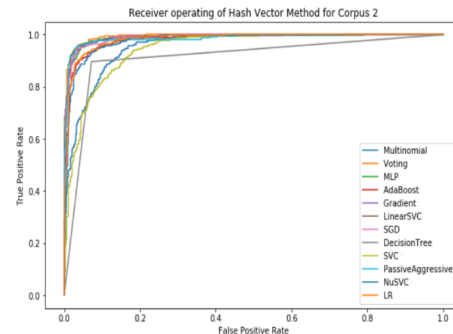
(e)



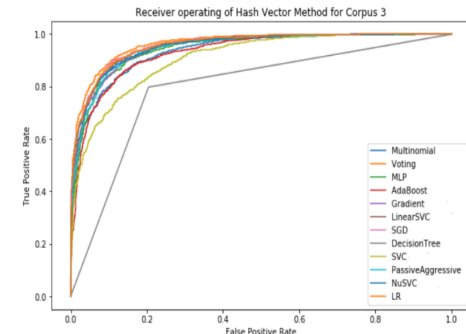
(f)



(g)



(h)



(i)

Figure (3.12) TPR vs FPR for CV on (a) News Trends, (b) Kaggle, (c) Reuters, Tf-Idf on (d) News Trends, (e) Kaggle, (f) Reuters, HV on (g) News Trends, (h) Kaggle, (i) Reuters datasets

3.5.1 Major findings

The major findings deal with the question, whether a trained classifier using old news articles can give accurate and efficient results to categorize the differences between fake and real content.

- It has been observed that the performance of classifying news article depends on the corpus and type of classification model.
- In our experiment, three corpus have been collected from three different sources [23] [24] [25]. The experiment was performed on these chosen datasets using Term frequency-Inverse document frequency (Tf-Idf), Count-Vectorizer (CV) and Hashing-Vectorizer (HV) feature extraction techniques.
- From the accuracy perspective, Passive Aggressive (93.2%) and LinearSVC (93.2%) outperform other models for all three (News Trends, Kaggle, Reuters) corpora. Whereas the Passive Aggressive (96%) and the LinearSVC (95.9%) performs best using Tf-Idf, Logistic Regression (94.9%) and Stochastic Gradient Descent (94.2%) performs best using CV, LinearSVC (90.6%) and Stochastic Gradient Descent (90.5%) performs best using HV individually for all three corpora.
- On evaluating F1-score metric, it was observed that Passive Aggressive (93.3%), Stochastic Gradient Descent (93.5%) and LinearSVC (93%) outperform other models on all three news article corpora using Tf-Idf, CV and HV feature extraction techniques.
- The Passive Aggressive (95.9%), Stochastic Gradient Descent (95.8%) and LinearSVC (95.4%) performs best using Tf-Idf, Logistic Regression (94.9%), Stochastic Gradient Descent (94.2%), Passive Aggressive

(93.7%), LinearSVC(93.2) performs best using CV and Passive Aggressive (90.4%), Stochastic Gradient Descent (90.7%) and LinearSVC (90.6%) performs best using HV.

- Passive Aggressive, Stochastic Gradient Descent and LinearSVC, Gradient Boosting, Logistic Regression outperform other ML models on basis of ROC_AUC metric with $TPR > 0.97$. Passive Aggressive, Stochastic Gradient Descent and LinearSVC gives $TPR > 0.97$ using Tf-Idf, Logistic Regression gives $TPR > 0.98$ using CV, Stochastic Gradient Descent, LinearSVC, Logistic Regression gives $TPR > 0.95$ for News Trends, Kaggle and Reuters dataset, respectively.
- The two ML classifiers such as Logistic Regression and LinearSVC are chosen among other models which results in both high accuracy and efficiency to overcome the trade-off issue.

Based on the training time required by various ML classifiers, it has been observed that there is a trade off between efficiency and accuracy. The training time required by HV is less than Tf-Idf and CV technique but it compromises accuracy metric. It has been analysed that the hashing technique is useful when the focus is to achieve high efficiency on a huge dataset. An ideal classifier should be proposed handle the trade off issue between efficiency and accuracy.

3.6 Proposed multi-level voting model

The proposed multi-level voting model not only helps to improve the accuracy but also helps to reduce the training time of the classifiers. The reduction in training time helps to increase the efficiency of our model by introducing

parallel methods where the base learners are generated parallelly. The motivation behind the proposed model is to analyse the independence between the base learners. Three levels are proposed to perform the experiment as shown in Figure 3.13.

Level 1: Sets of three ML classifiers are merged based on their performance metric (FP rate) to apply voting classifier. The voting models ($VC1, VC2, VC3$) are retrieved.

Level 2: A voting classifiers ($VC4$) is retrieved after merging the three models ($VC1, VC2, VC3$) on the basis of their false positive rate.

Level 3: The false predictions from voting classifier ($VC4$) are merged with PA and LinearSVC for Tf-Idf, LR and SGD for CV and SGD, LinearSVC for HV to get the final prediction.

On the basis of the minimum false positive (FP) rate, the ML models are merged using knowledge transfer from various datasets to overcome the weakness of existing individual models. The minimum is the FP ratio, the more accurate the model will be to predict the content as fake. Three feature extraction techniques (Tf-Idf, CV, HV) are used to extract the features from a collected dataset. Based on the FP ratio, the models are selected and merged to give an appropriate prediction. First cluster at level 1, SGD (News Trends), LR (Kaggle) and MLP (Reuters) using Tf-Idf; SGD (News Trends), LinearSVC (Kaggle) and MLP (Reuters) using CV; PA (News Trends), LR (Kaggle) and MLP (Reuters) using HV are merged to build the voting classifier ($VC1$). Second cluster at level 1, AdaBoost, Gradient Boosting and MultinomialNB (News Trends, Kaggle and Reuters) using Tf-Idf, CV and HV are merged to build $VC2$. Third cluster at level 1, SVC, NuSVC and DT (News Trends, Kaggle and Reuters) using Tf-Idf, CV and HV are merged to build $VC3$ as shown in Figure 3.13.

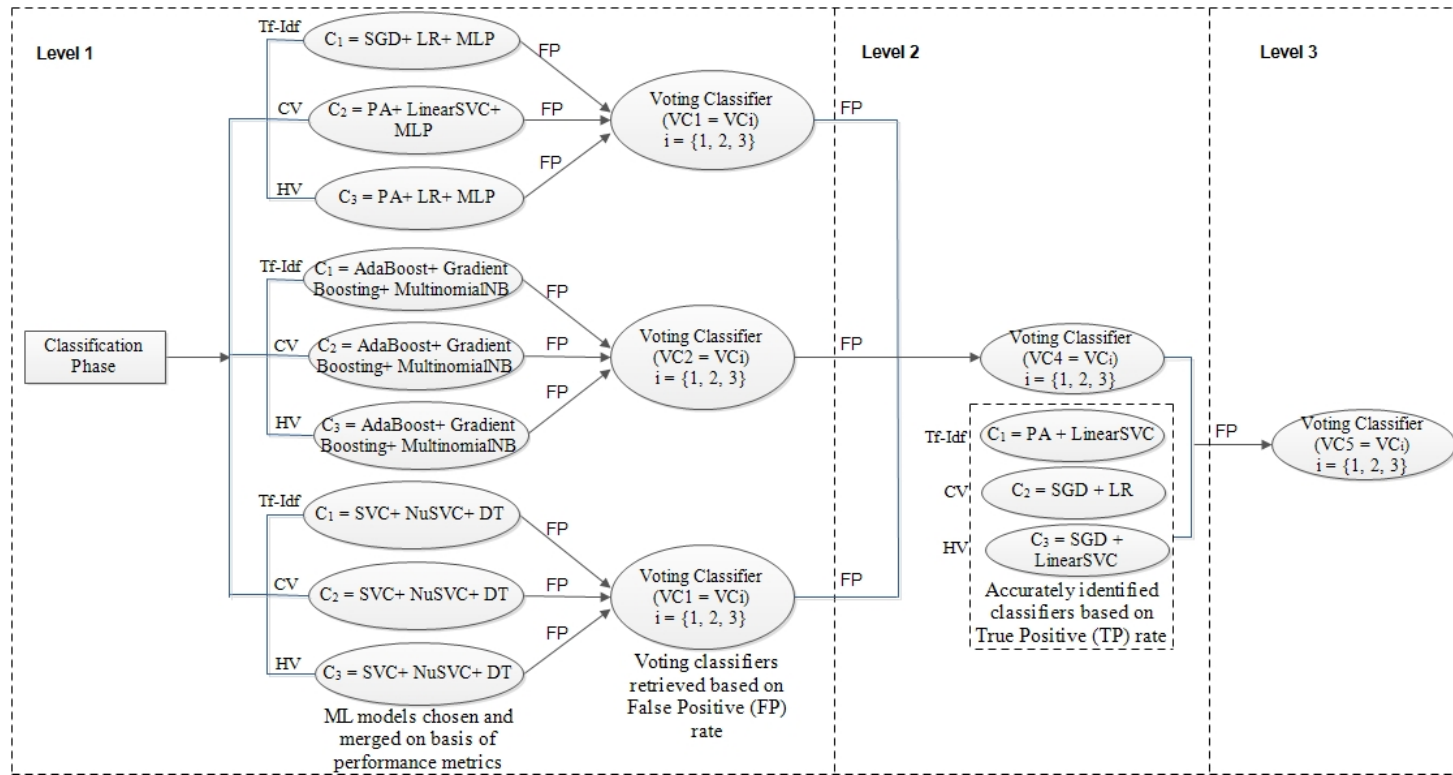


Figure (3.13) Architecture of the proposed multi-level voting model

Based on FP ratio of three proposed voting classifiers ($VC1, VC2, VC3$), a fourth voting classifier ($VC4$) is retrieved at level 2 based on the FP rates of $VC1, VC2$ and $VC3$ classifiers. At level 3, PA, LinearSVC using Tf-Idf; SGD and LR using CV; SGD and LinearSVC using HV are retrieved based on TP rate and are further clustered with $VC4$. Based on FP rate, $VC5$ is retrieved to give the final prediction of the proposed model.

3.7 Evaluating the performance of the proposed model

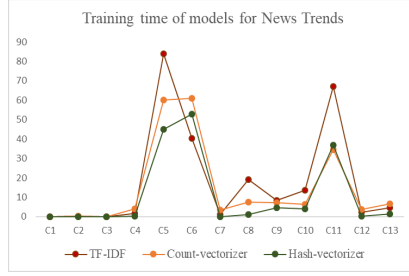
In the proposed multi-level voting classifier, the best three ML models are combined from each feature extraction technique as discussed in the previous section. It can be observed from Table 3.10 that the proposed model outperforms voting classifier by 0.73%, 0.66% and 0.13% using Tf-Idf, CV and HV, respectively in terms of accuracy metric. Similarly, the proposed model also gives significant improvement for precision, recall, specificity, and F1-score performance measures.

To compare the vectorization methods used in terms of training time required to train the data, Figure 3.14 depicts that the HV technique is more efficient than other feature extraction methods as the time required to train the proposed multi-level voting model is least for News Trends and Reuters dataset. To achieve better efficiency (less training time), Tf-Idf method is only suitable when data is not too huge whereas HV can be used for huge datasets when there is a requirement to make a trade-off between efficiency and accuracy.

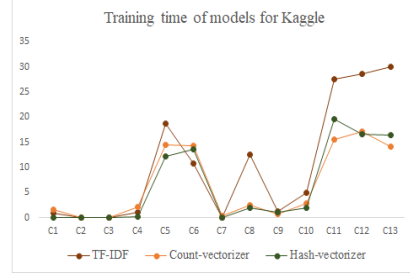
After performing our experiment, some conclusions are drawn in next

Table (3.10) Comparative analysis of the proposed multi-level voting model with voting classifier

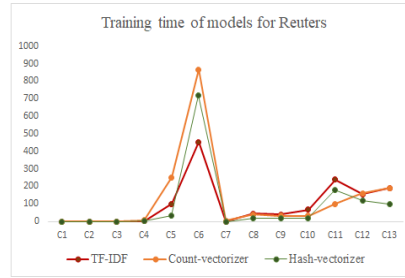
MODELS	News Trends			Kaggle			Reuters		
	Tf-Idf	CV	HV	Tf-Idf	CV	HV	Tf-Idf	CV	HV
	Accuracy								
Voting classifier	93.8	92.1	87.3	98.3	98.3	95.9	96.1	96.4	90.3
Multi-level voting classifier	94.3	93.6	87.1	98.9	98.7	95.8	97.2	96.5	90.2
	Precision								
Voting classifier	95.8	94.2	89.2	98.8	97.8	96.8	96.8	95	89.9
Multi-level voting classifier	96.4	94.3	89.6	99.1	98.3	96.8	98.4	97.6	90.4
	Recall								
Voting classifier	91.6	89.7	85.1	98	98.9	95.4	95	96.6	89.8
Multi-level voting classifier	93.1	91.4	85.2	98.7	98.8	94.4	96.8	95.2	90.8
	Specificity								
Voting classifier	95.9	94.3	89.5	98.6	97.6	96.3	96.4	94.6	88.8
Multi-level voting classifier	96	92.4	86.1	98.2	97.1	93.6	95.7	94.6	87.9
	F1-Score								
Voting classifier	93.7	91.8	87.1	98.3	98.3	96	95.8	95.7	89.8
Multi-level voting classifier	94.7	92.8	87.3	98.8	98.7	95.5	97.7	97	90.5



(a)



(b)



(c)

Figure (3.14) Training time comparison for (a) News Trends, (b) Kaggle and (c) Reuters dataset on basis of Tf-Idf, Hashing-Vectorizer and Count-vectorizer feature extraction techniques

section which helps to analyse the best classifier to be chosen for both high efficiency and accuracy.

3.8 Comparison with Existing Works

The results given by our system are also compared with other existing works on fake news detection as shown in Table 3.11, Table 3.12 and Table 3.13 using Reuters, Kaggle and News Trends corpora, respectively. Researchers have used ML approaches to perform fake news detection on various social media platforms.

Table (3.11) Comparison of existing models with a proposed system for Reuters corpus

Authors	Proposed Approach	Features	Model Accuracy
Cai <i>et al.</i> (2008)	Presented a Bayesian classification approach using class-specific features for automatic text classification.	Topic-based and document-based modelling.	LapPLSI - 74.6%
Mishu <i>et al.</i> (2016)	Classified text document using various classifiers	Document frequency	MultinomialNB- 72% LR- 73.5% SGD- 76% SVC- 78% LinearSVC- 83.3% Voting- 89%
Analysis based on individual ML models	Classified articles as fake or real using various classifiers.	TF-IDF	SGD- 96.2% PA- 96.2% LinearSVC- 96.3% MLP- 96.2% AdaBoost- 92.5% Voting- 96.4%
		Count-vectorizer	MLP- 96.2% DT- 88.5% LR- 95.7% MultinomialNB-89.9% Gradient Boosting- 92.5% SVC- 81.3%
		Hashing-vectorizer	NuSVC- 88.3%
Proposed Multi-level voting model	Fake news detection system is proposed to achieve high accuracy and high efficiency.	TF-IDF	Multi-level voting- 97.2%
		Count-vectorizer	Multi-level voting- 96.5%
		Hashing-vectorizer	Multi-level voting- 90.2%

Table (3.12) Comparison of existing models with a proposed system for Kaggle corpus

Authors	Proposed Approach	Features	Model Accuracy
Ahmed <i>et al.</i> (2017)	Proposed a fake news detection model using n-gram analysis and ML techniques	n-gram based TF and TF-IDF	LSVM-92% KNN- 83.1% SVM- 86% DT- 89% SGD- 89% LR- 89%
Analysis based on individual ML models	Classified articles as fake or real using various classifiers.	Count-vectorizer	MultinomialNB- 93.2% SVC- 52.9% LR- 96.4% MLP- 97.1% DT- 95.6% Voting- 98.3%
		TF-IDF	LinearSVC- 97.9% SGD- 98% PA- 98.3% AdaBoost- 97.3% Gradient Boosting- 98.5% Voting- 98.3%
		Hashing-vectorizer	NuSVC- 92.2%
Proposed Multi-level voting model	Fake news detection system is proposed to achieve high accuracy and high efficiency.	TF-IDF	Multi-level voting- 98.9%
		Count-vectorizer	Multi-level voting- 98.7%
		Hashing-vectorizer	Multi-level voting- 95.8%

It has been analysed from Table 3.11 that the feature extraction technique based on document frequency used by Mishu *et al.* (2016) for MultinomialNB (72%), SVC (78%) and voting classifier (89%) [149] does not give better accuracy than our proposed system for all three classifiers. In our proposed system, MultinomialNB (89.9%), SVC (81.3%), Voting (96.4%) outperforms Mishu *et al.* (2016) models.

It has been observed from Table 3.12 that the feature extraction technique based on Tf-Idf used by Ahmed *et al.* (2017) for ML classifiers does not gives better accuracy than our proposed system. The recorded observation shows

Table (3.13) Comparison of existing models with a proposed system for News Trends corpus

Authors	Proposed Approach	Features	Model Accuracy
Kuleshov <i>et al.</i> (2018)	Author shows the existence of adversarial examples in natural language classification	n-gram	NB- 93%
Analysis based on individual ML models	Classified articles as fake or real using various classifiers.	TF-IDF	PA- 93.5% SGD- 93.4% LinearSVC- 93.6% MLP- 93% DT- 81.3% AdaBoost- 86.7% Gradient Boosting- 89.2% Voting- 93.8% LR- 91.4%
		Count-vectorizer	MultinomialNB-89.3% SVC- 74.1%
		Hash-vectorizer	NuSVC-86.3%
Proposed Multi-level voting model	Fake news detection system is proposed to achieve high accuracy and high efficiency.	TF-IDF	Multi-level voting- 94.3%
		Count-vectorizer	Multi-level voting-93.6%
		Hash-vectorizer	Multi-level voting- 87.1%

that our proposed multi-level voting model (97.2%) outperforms Gradient Boosting by 0.4% using Tf-Idf, LR by 2.3% using CV and NuSVC by 3.6% using HV feature extraction techniques, respectively for Kaggle corpus. From Table 3.13, it can be analysed that our proposed multi-level voting model outperforms PA by 0.8% using Tf-Idf, MultinomialNB by 4.3% using CV and NuSVC by 0.8% using HV feature extraction techniques, respectively for News Trends corpus. From the comparative analysis, it has been analyzed that our proposed multi-level voting model using all three feature extraction technique outperforms when compared to the individual performance metrics of ML models used by Ahmed *et al.* (2017) and Mishu *et al.* (2016) for developing a system for automatic classification of news article features to

label them as fake or real news article. It has been observed from Table 3.11 that the proposed model outplays the PA model by 0.9% using Tf-Idf, LR model by 0.8% using CV and NuSVC model by 1.9% using HV feature extraction techniques, respectively when compared with their performance metrics.

3.9 Discussions

In this chapter, various machine learning classifiers were analysed to find the conditions under which particular model outperform others for identifying fake news articles. The set of attributes extracted from the corpus taken uses three feature extraction techniques (Tf-Idf, CV and HV) to feed the extracted feature vectors into the selected machine learning models. Some characteristics taken from datasets for learning task were categorical attributes, missing values, headlines of the article, the body of article and the publisher name. Various classifiers, MultinomialNB, PA, SGD, LR, SVC, NuSVC, LinearSVC, MLP, DT, AdaBoost, Gradient Boosting and Voting classifiers were analysed on basis of performance measures. After analysing the classifiers performance measures, the focus was to utilize the strengths of one model to complement the weakness of another. So, an ideal multi-level voting model was proposed, which integrates various ML models based on their FP rates to retrieve a news voting classifier to retrieve better prediction analysis. The developed model helps to solve the trade-off issue between accuracy and efficiency.

The proposed system is restricted to the datasets on which the model has been trained. It has been trained on the standard available datasets related to fake or real news articles. Model predicts accurately only for the English textual content and does not performs well on short-textual data.

The traditional techniques like Tf-Idf, CV and HV are used for feature extraction, which has become obsolete and in future, various word embeddings like *GloVe*, *Word2Vec*, *advance NLP techniques like phrases, named entities and shallow parsing can be used for making the proposed system more independent. To select the best approach among the borrowed machine learning models, no statistical tests were performed. The proposed method uses different feature extraction techniques so the qualitative comparison was not possible.*

In future, the proposed fake news article detection system can be used to classify the news article as fake or real at real-time on social media platforms like Facebook, Instagram, Twitter, *etc.* The proposed system has the potential to provide an impulse to various emerging applications such as controlling the spread of fake news during elections, terrorism, natural calamities, crimes for the betterment of the society.

Chapter 4

Clickbait Identification using Two-phase Hybrid CNN-LSTM Biterm Model

Clickbait refers to the content with a purpose to encourage visitors and attract attention by creating curiosity among users to click on a link to a particular web page [150, 151]. The maximum of the revenue generation for publishers of the online content revenue model is through advertising. Online content [6] attracts users through click links known as clickbaits by creating curiosity among visitors. The content shared on social media for advertising is in the form of short text, attachments including videos, audios, images, sharing links, *etc.* Clickbait is an umbrella term that encompasses all kinds of content capable of instigating an increased click-through [5]. Most of the clickbait headlines use the first form of a person like ‘I’, ‘We’, ‘You’. Some of the clickbait headlines are given below.

What we found was really shocking!

You can never guess what Happened!

What happens next will surprise you!

Just click to see this!

Did you know this!

OMG You won't believe your Eyes!

The primary danger posed by clickbaits is not only that the news topics such as economics, science, politics are replaced by business, sports, entertainment, but also the focus has shifted on attention-grabbing shareable contents. Often misleading and unverified headlines in the form of clickbaits are major contributors to the spread of fake news [152] on the Internet. In spite of all these effects of clickbaits, no comprehensive solution has been devised to remove the clickbait stories from the news feeds generated on user's accounts [153]. Currently, some solutions for automatic clickbait detection [154] are based on machine learning models [155], among which many of them yield low accuracy rates [156].

4.1 Contributions

A novel approach is proposed which is capable to identify the difference between legitimate and clickbait posts by directly analysing the headings for both textual and non-textual (text embedded in images) posts. To evaluate our model, three datasets have been collected. The first one is provided by Chakraborty *et al.* (2016) [4], second is provided by Khater *et al.* (2018) [157] and the third has been collected manually with the help of human annotations.

The main contributions are as follows.

- A ground dataset has been prepared from the Facebook page and Red-

dit website with the help of human annotations.

- Text extraction from the non-textual (image-based) data with the help of OCR tool using a pre-processing approach has been performed.
- Automatic identification of the eight types of clickbait headlines into *reasoning, number, reaction, revealing, shocking/unbelievable, hypothesis/guess, questionable, forward referencing* is done.
- A novel approach is proposed which works under a two-phase structure. In the first phase, the headlines of textual and non-textual data are fed to the CNN-LSTM model with the use of existing pre-trained vectors to identify whether the chosen post is legitimate or a clickbait. In the second phase, the classified clickbaits are fed to the existing Biterm Topic Model (BTM) which is a type of short text classifier. It uses a biterm co-occurrence of words to cluster similar clickbait headlines to identify the types of clickbait.
- A comparative study is performed with the existing systems to show the effectiveness of our proposed two-phase hybrid CNN-LSTM Biterm model.

4.2 Problem statement

The proposed work addresses two types of issues.

- To identify clickbait and non-clickbait headlines seen in the form of short text.
- To categorize the identified clickbait headlines into various clusters (topics).

Thus, the problem statement has been formulated as follows. To identify the headline as a binary classification problem, at the first phase the binary set of classes, $C = \{non_clickbait, clickbait\}$ are taken into consideration. Consider H as a set of all headlines and a training set $T \subseteq H \times C$ of labeled sentences to train a model to learn a function F . The function F maps the headlines H to $\{non_clickbait, clickbait\}$ such as $F : H \rightarrow C$. At the second phase, the clickbait headlines are classified into 8 categories, $C_t = \{Re, Nm, Rea, Rev, Sh, Hy, Ques, Fr\}$, where $Re \in Reasoning$, $Nm \in Number$, $Rea \in Reaction$, $Rev \in Revealing$, $Sh \in Shocking/Unbelievable$, $Hy \in Hypothesis/Guess$, $Ques \in Questionable$, $Fr \in Forward$ referencing using unsupervised model.

4.3 Data collection

Many sources are utilized for clickbait generation such as Twitter [64, 158], LinkedIn [10], Facebook [10], *etc.*, which are used by publishers as a trading platform. Such platforms are used to disseminate inappropriate information in the form of clickbait posts.

Various datasets are taken into consideration for evaluating our proposed model. The first dataset, taken from Chakraborty *et al.* (2016) [4] is named as *Dataset 1*. It consists of 32,000 headlines of news articles from several web domains such as ‘ViralStories’, ‘Scoopwhoop’, ‘Thatscoop’, ‘ViralNova’, ‘Upworthy’, ‘Buzzfeed’, ‘The Hindu’, ‘The Guardian’, ‘New York Times’, ‘Wikinews’. Among these headlines, a total of 15,999 were identified as clickbait headlines whereas 16,001 as non-clickbait headlines to develop a two-phase classification model.

The Second dataset, taken from Khater *et al.* (2018) [157] is named

as *Dataset 2*, which consists of 12,000 headlines. Among which 5,637 are the clickbaits and 6,080 are the non-clickbait headlines retrieved after pre-processing phase. ‘The Huffington Post’, ‘The Times Of India’, ‘NewsWeek’ and ‘BuzzFeed’ web domains were chosen to fetch the data for clickbait headlines whereas ‘The Indian Express’, ‘National Geographic’, ‘The wall street journal’, ‘The Economist’, ‘The Guardian’ and ‘The Hindu’ web domains were chosen to fetch the non-clickbait headlines.

A ground dataset has been created and is named as *Dataset 3*. The dataset was collected from two sources, *i.e.*, Reddit [27, 26] using Octoparse (a web-scraping tool) [159] and Facebook page [27] with the help of human annotations using weighted average score. Facebook and Reddit are the most popular social networking sites. The dataset was collected to analyse the distribution of both clickbait and non-clickbait headlines in terms of shares, likes, comments, domains, time (available from 1-DEC-2016 to 21-JUN-2019). The non-clickbait headlines were collected from subreddits [160, 161], which do not allow the clickbaits to creep in.

The overall statistics of our three collected datasets are discussed in Table 4.1, where the ratio of clickbait headlines for *Dataset 1*, *Dataset 2*, and *Dataset 3* are 49.99%, 46.97% and 40%, respectively. The vocabulary length to classify the headlines for *Dataset 1* are 18,966, for *Dataset 2* are 13,232 and 4,596 for *Dataset 3*.

Table (4.1) Statistics of collected datasets

Datasets	Total head- lines	Clickbait headlines	Non-clickbait headlines	Vocabulary length	Year of cre- ation
Dataset 1	32,000	15,999	16,001	18,966	2015
Dataset 2	12,000	5,637	6,080	13,232	2017
Dataset 3	1,800	1,200	600	4,596	2019

The length distribution of clickbait and non-clickbait headlines is repre-

sented by graphs shown in Figure 4.1. The X-axis labeled as ‘*text*’ represents the number of terms used in clickbait headlines whereas Y-axis represents the corresponding number of headlines having the same length distribution. It has been observed from Figure 4.1a and Figure 4.1b that the non-clickbait headlines have longer length distribution than clickbait headlines. Whereas from Figure 4.1c, no appropriate difference is noted between clickbait and non-clickbait headlines.

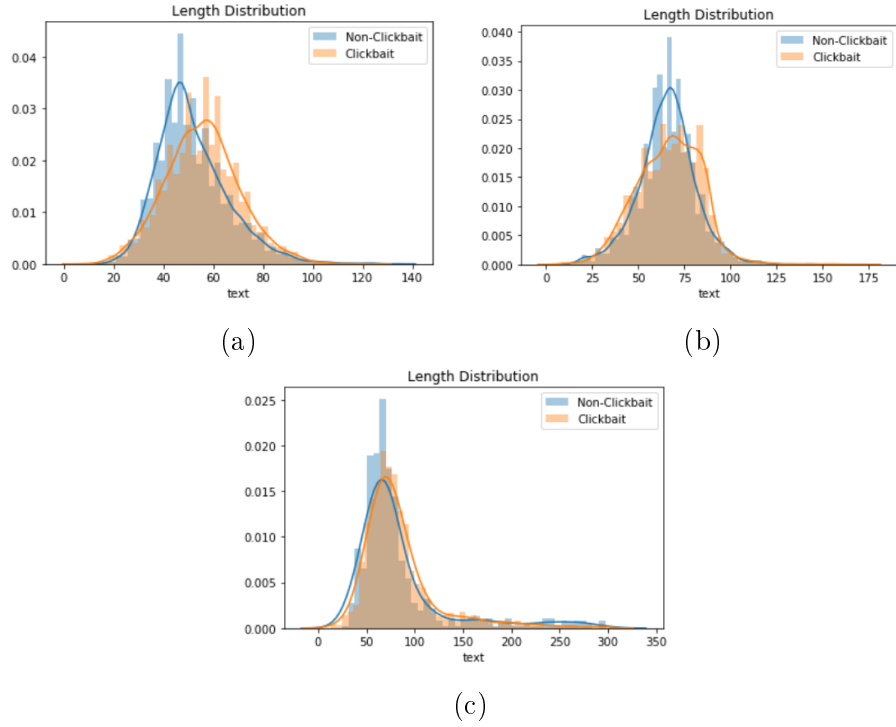


Figure (4.1) Length distribution of clickbait and non-clickbait headlines on (a) Dataset 1, (b) Dataset 2 and (c) Dataset 3

The mean distribution of the clickbait datasets as given in Table 4.2 for *Dataset 1*, *Dataset 2* and *Dataset 3* is 55.74, 66.72 and 84.27, respectively. The mean values of clickbait headlines are greater than the mean values

Table (4.2) Mean distribution of collected datasets

Datasets	Mean distribution of labeled datasets	Standard deviation of labeled datasets	Mean distribution of headlines labeled as clickbait	Mean distribution of headlines labeled as non-clickbait
Dataset 1	0.50	0.50	55.74	51.85
Dataset 2	0.51	0.49	66.72	65.65
Dataset 3	0.32	0.46	84.27	83.86

of non-clickbait headlines for the same datasets. The difference between the mean values is not so significant when evaluated on the basis of the Z-test. Hence, it can be concluded that the length distribution of headlines for clickbait news is often longer than the non-clickbait headlines published on social networking sites [73].

4.4 Architecture of the proposed system

The overview of the proposed two-phase model, where the hybrid CNN-LSTM model [162] is embedded at the first phase and the BTM [163] at the second phase to identify the type of online clickbaits is shown in Figure 4.2. To evaluate the proposed system, various datasets have been collected from three different sources in the first phase. During the pre-processing phase of textual data [164], extra white-spaces are removed and the conversion of all characters from upper to lower case is processed [165]. Whereas, the non-textual data is processed using the Optical character recognition (OCR) tool [166] to retrieve the textual content and then the pre-processing step is applied to it. Once completed with the pre-processing pipeline, the next step is to extract features [167] from the collected data. The extraction of features from the given data is called as feature engineering. It is done with the help of

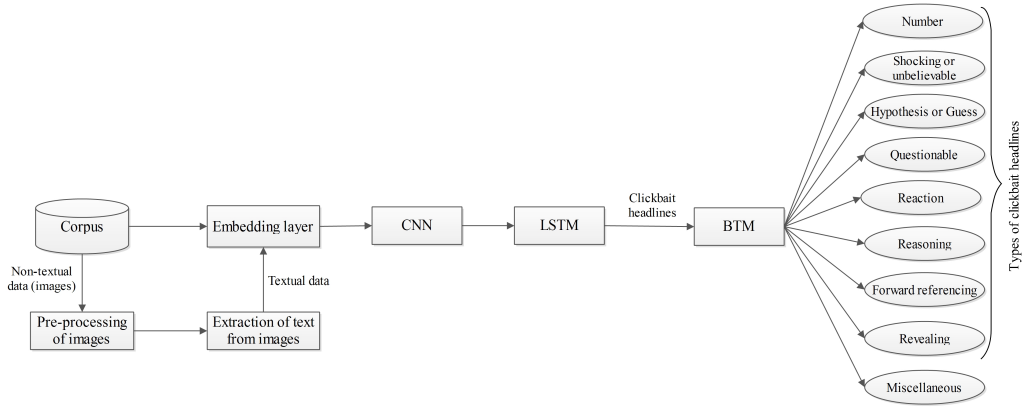


Figure (4.2) Overview of our proposed two-phase hybrid CNN-LSTM Biterm model for automatic detection and classification of various types of clickbait headlines

pre-trained embedding models. During the embedding process, various words are represented using dense vector representation. Word embedding is an improvement over the Bag of Words (BOW) approach [164] where each word is represented using sparse vectors to represent the entire vocabulary. The representations are sparse because every single word is represented by a row comprised of zero values. Whereas in an embedding layer every single word is represented by dense vectors. The dense vectors represent the projection of a word in a continuous vector space. The position of a word is represented within these vector spaces learned from the short headline text and is called as its embedding. In the next step, the retrieved vectors are fed to the hybrid CNN-LSTM model to classify the features of textual content. The model classifies the content as clickbait or non-clickbait. Further, the identified clickbait headlines are fed to the BTM (a short text classifier) to identify the type of clickbait [163]. The detailed description of each phase has been further discussed in this section.

4.4.1 Pre-processing

Our system can take two types of input, *i.e.*, in the form of textual and non-textual data as shown in Figure 4.3. To process textual data there is no

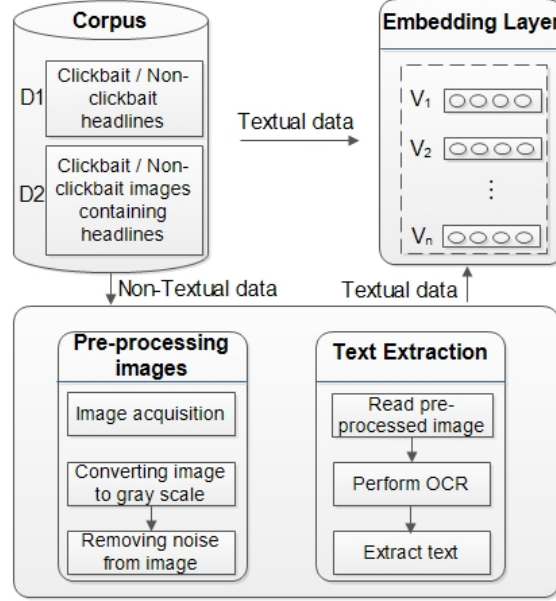


Figure (4.3) Pre-processing of non-textual (image-based) data to extract textual content

requirement of OCR tool. Whereas non-textual data cannot be directly processed and fed to the proposed model. So, a pre-processing step is required to extract the textual information from non-textual (image-based) data. During the pre-processing of non-textual data, following operations are performed.

- **Converting to grayscale:** The quality of the image is improved by removing the color variation by converting the original image (as shown in Figure 4.4a) into a grayscale image (as shown in Figure 4.4b) for accurate text detection.

- Noise removal: Non-local Means Denoising algorithm [168] has been used for noise removal from an image. The denoised image retrieved can be seen in Figure 4.4c.
- Extraction: The Google cloud vision (an OCR tool) is used to extract text from images. It requires Google cloud access and the creation of an API key. The *google-cloud-vision* and *google-cloud* libraries have been used for reading the images for further processing [166].
- Auto-correction: The Python libraries like *autocorrect* and *language_check* are used in the experiment to correct the spelling errors [169].

Whereas during the pre-processing phase of textual headlines, all the missing values are removed from the document using *dropna* method in Python dataframe, *lower* and *sub* methods are used to convert the upper characters to lower cases and remove the whitespaces, respectively.



Figure (4.4) Pre-processing on (a) Original image, (b) Grayscale image and (c) Denoised image

The formal representation of features chosen from textual and non-textual data has been discussed below. Let K be the collection of posts published online over social media platforms and H is defined as a collection of headlines

Table (4.3) Embedding layer specifications of the proposed model

Arguments	Description	Size
Input Dimension	It specifies the size of vocabulary in our collected corpus	20,000
Output Dimension	It specifies the size of vector space where words are embedded	128
Input length	It specifies the length of the input sequences, <i>i.e.</i> , the maximum limit of words for each headline.	100

extracted from posts K . $H(i)$ is defined as the extracted headline from the i^{th} post as represented by Eq. (4.1),

$$H(i) = \begin{cases} K(i), & \text{if } K(i) = \text{Textual data} \\ OCR(K(i)), & \text{if } K(i) = \text{Non - Textual data} \end{cases} \quad (4.1)$$

where i is the index which is repeated for the collection of posts with range $[0 : (|K| - 1)]$. $OCR(K(i))$ is considered as a function that extracts text from the image if the textual content is present else stores *NULL*.

The embedding layer used in the proposed model is discussed in the following section.

4.4.2 Embedding layer

Word embeddings are used to provide a better vector feature representation of words. *Gensim* (NLP library) provides an amazing wrapper to adopt different pre-trained word embedding models which include *GloVe* (by Stanford) [170] and *Word2Vec* (by Google) [171].

The embedding layer specifies three arguments that are defined for the first hidden layer of the hybrid CNN-LSTM model as shown in Table 4.3.

GloVe

Global Vectors for Word Representation (*GloVe*) is provided by the Stanford NLP team. The team also provides various versions of the *GloVe* pre-trained vectors. Among various available versions, *GloVe 1.2* is used to perform the experiment. It consists of four files with four different embedding representations such as *glove.6B.50d.txt* with 6 billion tokens and 50 features, *glove.6B.100d.txt* with 6 billion tokens and 100 features, *glove.6B.200d.txt* with 6 billion tokens and 200 features, and *glove.6B.300d.txt* with 6 billion tokens and 300 features [170]. The script uses *glove.6B.100d.txt* (with 400,000 vocabulary size) file to train the hybrid CNN-LSTM model. The *GloVe* is an unsupervised learning model that is used to generate word to word co-occurrence matrix from a corpus. Python library named as *glove_python* is used to implement *GloVe* embeddings.

Word2Vec

It is a type of predictive model which is used to predict the destination word from the context of its neighboring words. The word vectors are trained on Google news data using skip-gram to build the model provided by [171] in 2013. Each word is encoded using a one-hot encoding scheme and then is fed to the hidden layer of the hybrid CNN-LSTM model using a matrix of weights. Our experiments also used a pre-trained vector with a vocabulary size of 3 million words trained from 100 billion words of Google News data. The pre-trained vector file for 300-dimensional word vectors was downloaded to perform the experiment.

4.4.3 Deep-learning based hybrid classifier

The architecture of the hybrid classifier mainly consists of two components, *i.e.*, Convolutional Neural Network (CNN) and Long-Short Term Memory (LSTM). CNN [172] is applied to extract high-level sequences of word features. Whereas the LSTM [173] is embedded to capture long term sequences and also reduces the training time of the proposed model. The processed data (textual and non-textual) is fed to the hybrid CNN-LSTM model [162] where hyper-parameters are tuned in a standard split (training (80): testing (20)).

CNN component

The data extracted from the embedding layer is fed to the CNN module. Where the features are detected at different regions with the help of a sliding filter vector evolved in the convolution layer. Consider $u_j \in \mathbb{R}^n$ as the n -dimensional word vectors of a particular word in a headline for j^{th} position. Let $u \in \mathbb{R}^{L \times n}$ be the input headline, where L denotes the length of the headline. Consider f as the length of the filter and vector $v \in \mathbb{R}^{f \times n}$ be the filter for performing convolution operation. A window vector w_i with f consecutive word vectors, where i is the position of a word in a headline is given by Eq. (4.2).

$$w_i = [u_i, u_{i+1}, \dots, u_{i+f-1}] \quad (4.2)$$

Where commas denote the row vector concatenation. The filter v revolves at each position with the window vectors to generate a feature map $m \in \mathbb{R}^{L-f+1}$, where each element m_i of feature map for window vector w_i is represented by Eq. (4.3).

$$m_i = t(w_i \odot v + b) \quad (4.3)$$

Where b is the biased term that belongs to $\mathbb{R}$ and t is the non-linear transformation function and $\odot$ is the element-wise multiplication. In our experiment, *ReLU* [174] is chosen as a non-linear function. To generate multiple feature maps, the hybrid CNN-LSTM model uses multiple filters. For each row W_i of $W \in \mathbb{R}^{(L-f+1) \times z}$ is a new feature representation generated from z filters for window vector at position i . Max-pooling is applied to feature maps after the convolution layer which helps to select the top most important features. Further, the successive window representation is fed into the LSTM model.

LSTM component

In LSTM, the output of the module is controlled by a set of gates (forget gate (f_s), input gate (i_s), output gate (o_s)). In our experiment, all vectors used in LSTM architecture share the same memory dimensions. The working of i_s , f_s and o_s is denoted by Eq. (4.4), Eq. (4.5) and Eq. (4.6).

$$i_s = \sigma(W_i \cdot [h_{s-1}, x_s] + b_i) \quad (4.4)$$

Where σ is the logistic sigmoid function that gives output between $[0,1]$, x_s is the input at current time step s and h_{s-1} is the previous output of the hidden state.

$$f_s = \sigma(W_f \cdot [h_{s-1}, x_s] + b_f) \quad (4.5)$$

Here, f_s is the function to control the extent of information required from the old memory cell to be thrown away.

$$o_s = \sigma(W_o \cdot [h_{s-1}, x_s] + b_o) \quad (4.6)$$

Here, o_s is the control of output on the memory cell. Current input and previous output of the hidden state are taken into consideration to update the

current hidden state h_s and current memory cell c_s through various transition functions as shown by Eq. (4.7), Eq. (4.8) and Eq. (4.9),

$$q_s = \tanh(W_q \cdot [h_{s-1}, x_s] + b_q) \quad (4.7)$$

where $\tanh$ is the hyperbolic function that gives output between $[-1,1]$.

$$c_s = f_s \odot c_{s-1} + i_s \odot q_s \quad (4.8)$$

Here, c_s represents the current memory cell and $\odot$ denotes the element-wise multiplication,

$$h_s = o_s \odot \tanh(c_s) \quad (4.9)$$

where h_s represents the current hidden state.

LSTM model is chosen and embedded on the CNN model to learn the long term sequence features. To get the final output from the hidden layer, sigmoid layer is used.

Implementation of hybrid classifier

The number of filters used in the hybrid CNN-LSTM model for our experiment is 128. The size of 1-dimensional convolution window is considered as 3 and the *ReLU* activation function is used for the hidden layers. The value of the pool size is selected as 3 for max-pooling layers used in the hybrid CNN-LSTM model [162]. The early stopping method [175] is used to decide the number of training epochs. Initially, 50 epochs were chosen to analyse the model performance. It was observed that the model performance stops showing any significant improvement after 15 epochs as the accuracy rate remains nearly the same till 50 epochs. So, during the learning process, our model is trained for 15 epochs. The architecture of the hybrid CNN-LSTM model is shown in Figure 4.5.

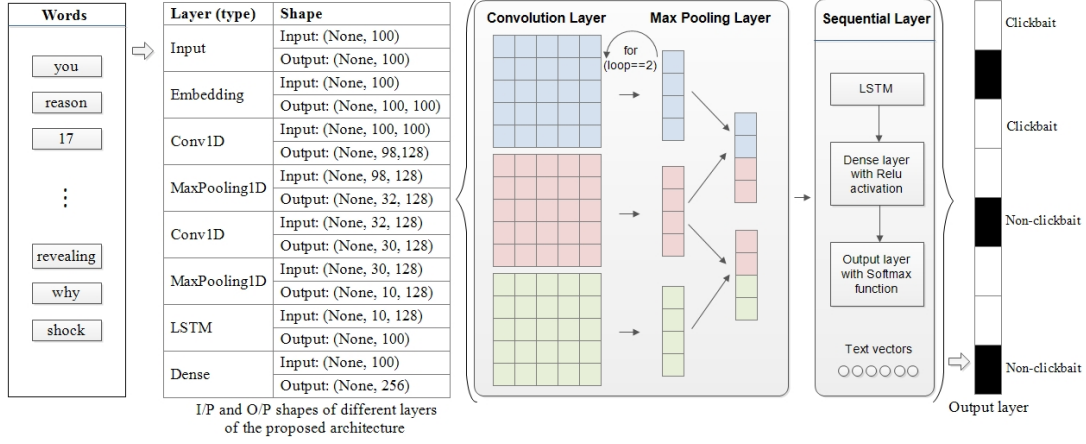


Figure (4.5) Architecture of the hybrid CNN-LSTM model

The input size in this architecture is the maximum sequence length chosen for clickbait headlines, *i.e.*, 100. The input is fed to the embedding layer. The output dimensions (100 x 100) are retrieved from the embedding layer which is further fed to the convolution layer, where the number of parameters used is 38,528. The output dimensions of the first convolution layer are 98 x 128, which is fed as an input to the first max-pooling layer giving an output of 32 x 128. The second convolution layer takes 32 x 128 dimensions as input and trains the vector with 49,280 parameters. In the next step, the second max-pooling layer takes 30 x 128 input dimensions from the second convolution layer and gives 10 x 128 as an output shape. The output from max-pooling layer is fed to LSTM (flattening) layer which gives 100 values with 91,600 parameters. The output is then fed to the dense layer with 256 (values) x 128 (number of neurons) + 128 (bias values for neurons) along with 25,856 number of parameters, which is fed to the sigmoid layer to identify the clickbait headlines.

The total parameters, trainable and non-trainable parameters taken by

Table (4.4) Number of parameters in the hybrid CNN-LSTM model using *GloVe* embeddings

Corpus	Total Parameters	Trainable parameters	Non-trainable parameters
Dataset 1	2,287,701	287,701	2,000,000
Dataset 2	219,422	217,722	17,000
Dataset 3	1,268,501	268,501	1,000,000

the hybrid CNN-LSTM model for *Dataset 1*, *Dataset 2* and *Dataset 3* are shown in Table 4.4. It has been observed that the total parameters of our collected dataset are 1,268,501, whereas only 268,501 parameters are used by the model to find the optimal weights to reduce the cost function of the model.

The classified clickbait headlines retrieved from the hybrid CNN-LSTM model are fed to BTM to identify the types of clickbait.

4.4.4 Biterm Topic Model (BTM)

Discovering topics from short texts has genuinely become an intractable problem. It was hard for traditional topic models such as *k-Means*, *Probabilistic latent semantic analysis (PLSA)*, *Latent discriminant analysis (LDA)* [176] to identify short texts as they suffered from severe data sparsity problem when implemented on short texts. So, the BTM came into existence to overcome this limitation.

BTM is a word co-occurrence biterm based model as shown in Figure 4.6. Where a biterm consists of two words co-occurrence in the same headline. It is a type of generative model which generates a biterm by making a two word pattern from the same topic t . The number of topics (t) chosen to apply BTM was decided after analysing the elbow curve [177]. A plot was drawn by taking into consideration the appropriate number of topics. The point of a

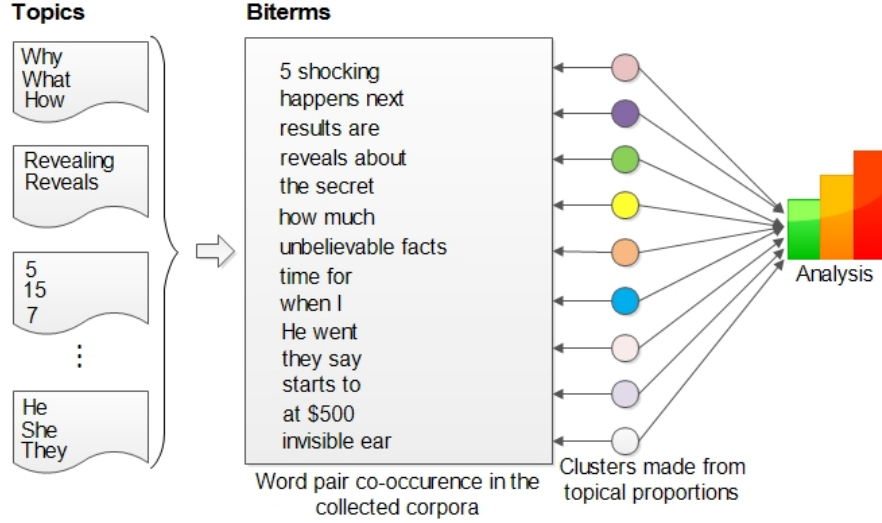


Figure (4.6) Architecture of the Biterm Topic Model (BTM)

bend (known as knee) was tested by varying the value of t from 4 to 12. The best compactness of clustering was considered when the value of t was set to 9, where an appropriate bend of the curve was seen. The first eight clusters (topics) define the types of clickbait and the last cluster (9^{th}) defines the miscellaneous clickbait headlines as it was a loosely packed cluster covering mixed topics. Some of the generative processes for the collected corpus in BTM where β and γ are considered as the Dirichlet priors (Dp) [163] is discussed as follows.

- For each topic t ,
 - a topic-specific word distribution is drawn $\delta_t \sim Dp(\gamma)$
- Draw a topic distribution $\vartheta \sim Dp(\beta)$ for the chosen dataset
- For every biterm (b) in biterm set (B)
 - draw a topic assignment $t \sim Multinomial(\vartheta)$

- draw b (*i.e.*, two word occurrence from the same headline):
 $(w_x, w_y) \sim \text{Multinomial}(\delta_t)$

The joint probability of b can be represented by Eq. (4.10).

$$P(b) = \sum_t P(t)P(w_x|t)P(w_y|t) = \sum_t \vartheta_t \delta_{x|t} \delta_{y|t} \quad (4.10)$$

Hence, the likelihood of the whole chosen dataset is represented by Eq. (4.11).

$$P(B) = \prod_{(x,y)} \sum_t \vartheta_t \delta_{x|t} \delta_{y|t} \quad (4.11)$$

The argument t represents the number of topics chosen to classify the types of clickbait. The symmetric Dirichlet prior probability of a topic is $P(t)$ and is kept 1 whereas the symmetric Dirichlet prior probability of a word with given topic $P(w|t)$ is kept to 0.01. The Gibbs algorithm [178] is used for categorizing the type of clickbait, the number of iterations for Gibbs sampling is kept 10. The default window size (15) is considered for performing the second phase experiment.

Thus, the BTM helps to convey the semantic features of related topics by making use of co-occurrence patterns based on biterms.

4.5 Experimental results

The configuration of the computing environment includes: Processor is Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz 2.60GHz, DDR3 16GB RAM and Nvidia Titan Xp GPU with 3840 Cores running at 1404MHz. Theano 1.0.3 [179] is used with Python version 3.6.3 on Windows 10 Pro Operating System. To evaluate the proposed model, the performance analysis of each phase has been discussed in this section.

Table (4.5) Results obtained after performing OCR and pre-processing on Dataset 3

Tool	Total images	Correctly identified	Incorrectly identified	Accuracy	Precision	Recall	F1-score
Google cloud vision	1,800	1,578	222	87.60	91.34	86.56	88.88
Google cloud vision after pre-processing	1,800	1,601	199	88.94	92.7	89.6	91.12

4.5.1 OCR performance

It has been observed from the results obtained in Table 4.5 that the Google cloud vision tool [166] gives an accuracy of 87.60% while extracting headlines from the non-textual (image-based) posts for our *Dataset 3*. After pre-processing the images, it was analysed that the accuracy to recognize correct text from images increased by 1.34%. Also, it has been observed that the Google cloud vision tool performs best on *.bmp*, *.tiff*, *.png*, *.jpg* and *.jpeg* to extract text from images [180]. The collected *Dataset 3* used for the experiment has only *.jpg* and *.png* formats.

Tag clouds

Tag clouds of top 12 words are retrieved after pre-processing the headlines as shown in Figure 4.7, Figure 4.8 and Figure 4.9.

Figure 4.7a shows the top words like *guess*, *questions*, *reasons*, *you*, *identify* retrieved from the clickbait headlines, whereas top words retrieved from non-clickbait headlines like *inauguration*, *iraq*, *advantage*, *government*, *supports* are shown in Figure 4.7b for *Dataset 1*. Top words like *Twitter*, *celebs*, *actually*, *quote*, *most*, retrieved from the clickbait headlines are depicted in Figure 4.8a and words like *delhi*, *family*, *mumbai*, *england*, *uk* are seen among top words, as shown in Figure 4.8b for *Dataset 2*.



Figure (4.7) Top 12 (a) Clickbait and (b) Non-clickbait word clouds generated from Dataset 1



Figure (4.8) Top 12 (a) Clickbait and (b) Non-clickbait word clouds generated from Dataset 2

For our collected *Dataset 3*, the tag clouds includes *why*, *don't*, *guess*, *what*, *reason*, *secret*, *worse* top words for clickbait headlines, as depicted in Figure 4.9a and words like *saudi*, *agreement*, *sumbit*, *bomb*, *israel*, *india*, *pakistan*, *russia* are retrieved from non-clickbait headlines, as demonstrated in Figure 4.9b.

It can be concluded that the top words extracted from the clickbait headlines create the curiosity among users and urge them to click to view such type of headlines seen on various social media platforms in form of posts. The words retrieved from the non-clickbait headlines are genuine and do not create any type of confusing statements among users.



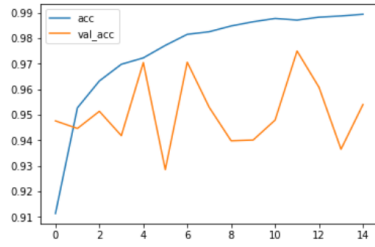
Figure (4.9) Top 12 (a) Clickbait and (b) Non-clickbait word clouds generated from Dataset 3

4.5.2 Classification performance

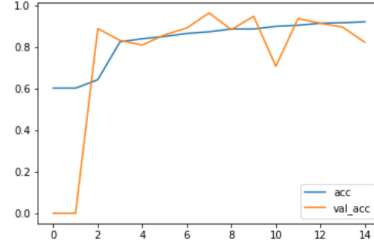
In this experiment, classification has been performed on *Dataset 1*, *Dataset 2* and *Dataset 3*. To validate our proposed model various results are discussed in this section. It has been observed from Figure 4.10 that the validation loss value for *Dataset 1*, *Dataset 2* and *Dataset 3* is 0.18, 0.43 and 0.27, respectively.

The performance analysis of three models (hybrid CNN-LSTM without pre-trained vectors, hybrid CNN-LSTM with *Word2Vec* and *GloVe* pre-trained vectors) is evaluated in terms of accuracy, precision, recall, F1-score and AUC-ROC as depicted in Table 4.6. It has been observed that the hybrid CNN-LSTM model with pre-trained embeddings (*Word2Vec* and *GloVe*) performs better than the hybrid CNN-LSTM model without pre-trained embeddings for all three datasets.

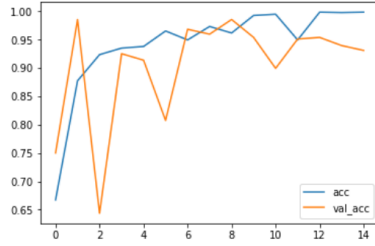
The accuracy metric of the hybrid CNN-LSTM model using *GloVe* embeddings achieves an increase of 2.38%, 4.57% and 4.25% for *Dataset 1*, *Dataset 2* and *Dataset 3*, respectively, when compared with the hybrid CNN-LSTM model using *Word2Vec* embeddings. Also, an increase of 3.9% in precision and 5.74% in recall values is seen for our collected *Dataset 3*. It



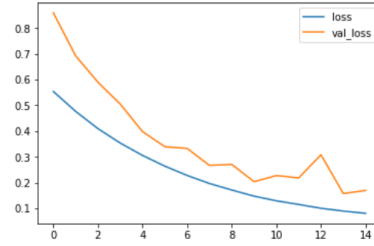
(a)



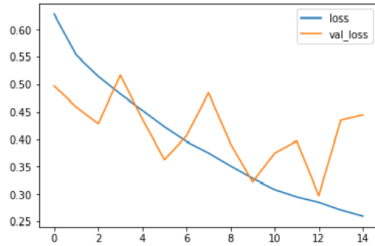
(b)



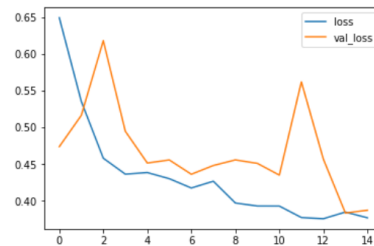
(c)



(d)



(e)



(f)

Figure (4.10) Training accuracy vs validation accuracy for (a) Dataset 1, (b) Dataset 2, (c) Dataset 3, and Training loss vs validation loss for (d) Dataset 1, (e) Dataset 2, (f) Dataset 3 at different epochs

Table (4.6) Comparative analysis of the hybrid CNN-LSTM architecture with and without pre-trained vectors using performance metrics

Methods	Dataset	Accuracy	Precision	Recall	F1-score	AUC-ROC
Without Pre-trained vectors	Dataset 1	81.85	82.89	79.76	81.29	0.87
	Dataset 2	78.48	80.25	78.12	79.17	0.81
	Dataset 3	84.61	84.7	80.53	82.56	0.87
With <i>Word2Vec</i> pre-trained vectors	Dataset 1	93.42	92.34	80.01	85.73	0.97
	Dataset 2	84.87	90.00	81.33	85.44	0.89
	Dataset 3	89.96	91.97	85.66	88.70	0.94
With <i>GloVe</i> pre-trained vectors	Dataset 1	95.8	91.24	85.43	88.23	0.99
	Dataset 2	89.44	95.64	89.69	92.56	0.94
	Dataset 3	94.21	95.87	91.4	93.58	0.98

can be observed from Table 4.6 that the hybrid CNN-LSTM model when pre-trained with the *GloVe* embeddings outperforms the hybrid CNN-LSTM model when pre-trained with *Word2Vec* embeddings in terms of accuracy, recall, F1-score, AUC-ROC performance metrics. Hence, the hybrid CNN-LSTM model with pre-trained *GloVe* embeddings is used to further classify clickbait and non-clickbait headlines.

The classified clickbait headlines are fed to BTM with specified arguments considered in our experiment.

4.5.3 Cluster analysis to identify the types of clickbait headlines

The approach followed in this work offers a preliminary discussion on misleading clickbait headlines. Various characteristics of misleading clickbait headlines have been analysed in this section.

To decide the appropriate number of topics (t) to apply the BTM, an elbow method [177] was applied. After running the elbow algorithm by varying the size of t from the range 4 to 12, it was analysed that the best compact-

Table (4.7) Coherence score of top 5, 10 and 20 words for Dataset 1, Dataset 2 and Dataset 3 among nine clusters retrieved using BTM to analyse the quality of clusters

Clusters	Top words								
	Dataset 1			Dataset 2			Dataset 3		
	5	10	20	5	10	20	5	10	20
<i>Cluster₁</i>	-16.31	-93.89	-302.17	-15.71	-109.21	-428.83	-5.67	-60.67	-371.41
<i>Cluster₂</i>	-19.01	-97.28	-425.67	-20.78	-102.9	-432.01	-14.55	-89.07	-330.38
<i>Cluster₃</i>	-20.28	-76.09	-287.97	-20.05	-94.11	-445.36	-16.48	-81.62	-336.96
<i>Cluster₄</i>	-20.03	-98.64	-353.91	-22.7	-93.19	-444.3	-17.34	-63.33	-376.81
<i>Cluster₅</i>	-20.63	-105.92	-362.89	-24.67	-92.23	-455.63	-19.69	-85.37	-348.38
<i>Cluster₆</i>	-21.55	-103.23	-379.1	-21.91	-118.02	-459.39	-17.54	-63.38	-360.41
<i>Cluster₇</i>	-22.13	-107.39	-389.15	-21.8	-116.13	-469.85	-18.55	-69.06	-380.7
<i>Cluster₈</i>	-22.46	-85.1	-430.3	-27.11	-109.21	-428.83	-22.44	-83.63	-349.88
<i>Cluster₉</i>	-25.53	-108.56	-446.45	-29.81	-121.45	-474.4	-28.18	-91.09	-376.84

ness of clustering was seen at 9. Thus the number of topics selected to apply BTM was 9.

To evaluate the topic quality of clusters, an automated metric, *i.e.*, coherence score has been used [181]. The coherence score of nine clusters for *Dataset 1*, *Dataset 2* and *Dataset 3* is shown in Table 4.7. Three samples of top words (5, 10 and 20) are used to analyse the coherence score of each topic for *Dataset 1*, *Dataset 2* and *Dataset 3*. It was observed that for each topic, some specific common words for all three collected samples of top words (5, 10 and 20) were extracted. Some frequent occurrence of similar words like *anger*, *guilt*, *react*, *reply* were seen in *Cluster₃* and words like *unbelievable*, *impossible*, *shock*, *wow* were seen in *Cluster₅*, where the coherence score signifies the measure of the occurrence of frequent words in a topic. It was observed from Table 4.7 that the quality of *Cluster₉* was very poor for our collected *Dataset 1*, *Dataset 2* and *Dataset 3*. The coherence score for top words (5, 10 and 20) for *Dataset 3* is -28.18, -91.09 and -376.84, respectively.

The more positive coherence score depicts that the topics are more coherent and have better quality.

Various types of clickbait headlines are defined with an appropriate example in Table 4.8.

Table (4.8) Description of identified topics

Topics	Priority	Definition	Example
Reasoning	5	deducing the fact out of the statement	here's why chelseamperetti left "brooklyn nine-nine".
Number	8	creating curiosity by a particular quantum/figure	29 pictures that will ruin your whole dang day.
Reaction	3	a process to act in response to a situation	rumour: next DLC fighter for smash bros. ultimate seemingly teased by game supervisor.
Revealing	7	making interesting by allowing a look at hidden content	star trek: william shatner reveals if he'd return as kirk for a new series.
Shocking/Unbelievable	2	causing a sudden violent impact or blow in mind	Man tries to hug a wild lion, you won't believe what happens next
Hypothesis/Guess	1	committing oneself to an opinion with little knowledge	you'll never guess what language kendall jenner can speak.
Questionable	4	trying to invite inquiry	can you shoot a feature film on an iPhone?
Forward Referencing	6	use of demonstratives, pronouns to create a reference to entities	firefighters rescued litter of puppies. then they realized they weren't actually dogs.

The distribution of clickbait headlines in each categorized cluster is shown in Table 4.9. It can be observed that 14,542 out of 15,999 clickbait headlines from *Dataset 1* have been categorized among 8 types of clusters while the

remaining headlines are clustered as a ‘Miscellaneous’ set (9th cluster) of clickbait headlines. The count of clickbait headlines from *Dataset 2* is 5,637

Table (4.9) Distribution of headlines

	Dataset 1			Dataset 2			Dataset 3		
Types of clickbait	Non-clickbait	Clickbait	Clickbait %	Non-clickbait	Clickbait	Clickbait %	Non-clickbait	Clickbait	Clickbait %
<i>Cluster</i> ₁ (Reasoning)	18	257	93.45	12	66	84.61	4	50	92.5
<i>Cluster</i> ₂ (Number)	2,981	7,057	70.3	3,875	3,248	45.59	28	482	94.50
<i>Cluster</i> ₃ (Reaction)	8	43	84.31	1	24	96	0	5	100
<i>Cluster</i> ₄ (Revealing)	45	94	67.6	20	27	57.44	7	63	90
<i>Cluster</i> ₅ (Shocking or Unbelievable)	1	130	99.23	9	39	81.25	0	16	100
<i>Cluster</i> ₆ (Hypothesis or Guess)	10	226	95.76	2	32	94.11	0	29	100
<i>Cluster</i> ₇ (Questionable)	290	3,544	92.43	190	1,238	86.69	18	196	91.58
<i>Cluster</i> ₈ (Forward referencing)	200	3,191	91.96	250	530	67.94	102	330	76.38
Sub-total	3,553	14,542	-	4,359	5,204	-	159	1,171	-
<i>Cluster</i> ₉ (Miscellaneous)	12,448	1,457	10.47	1,721	433	20.1	441	29	6.17
Total	16,001	15,999	-	6,080	5,637	-	600	1,200	-

from which 5,204 have been categorized into 9 clusters using BTM.

Dataset 3 has 1,200 clickbait headlines, among which 1,171 have been categorized into 8 clusters. It has been observed from Table 4.9 that the clickbaits such as *Shocking/Unbelievable*, *Hypothesis/Guess* and *Reaction* are the highest in numbers among rest of the clickbait headlines published over social media platforms. Based on these findings, a priority number has been assigned to the identified clusters in Table 4.8. For the priority attribute, value 1 signifies the highest priority whereas 8 signifies the lowest priority.

The repeatable code for our proposed model has been uploaded to GitHub¹. It will help the research community to regenerate the results.

4.6 Comparison with existing systems

Our proposed model outperforms other existing clickbait detection systems as shown in Table 4.10. The dataset created for clickbait and non-clickbait headlines in this work has been collected from various social media platforms (Reddit, Facebook). The collected dataset consists of textual and non-textual (images with clickbait headlines) data. Agrawal and Amol (2016) also created the ground corpus (only textual data) for detecting clickbaits online using Reddit, Facebook, and Twitter platforms [27]. They used CNN with one layer convolution for detecting clickbaits using two variants of embeddings. The first variant includes *click-scratch* features which give 89% accuracy with 0.88 precision, 0.80 recall and 0.84 F1-score values. The second variant includes *click-Word2Vec* features which give 90% accuracy with 0.85 precision, 0.88 recall and 0.86 F1-score values.

¹<https://github.com/sawinderkaurvohra/Clickbait-Detection>

Table (4.10) Comparison of the experimental results of our proposed model with the existing systems

Author	Proposed approach	Feature selection	Models	Dataset	Results
López-Sánchez <i>et al.</i> (2017) [182]	The proposed approach uses metric learning and deep learning algorithms by integrating them with Case-Based Reasoning methodology.	Tf-Idf, n-gram, 300 dimensional <i>Word2Vec</i>	CBR + CNN	Largest collected publicly available dataset Chakraborty <i>et al.</i> (2016) [4] having 32,000 (clickbait and non-clickbait) headlines.	The proposed approach achieved 0.994, 0.95, 0.90 average area under the ROC curve using <i>Word2Vec</i> , <i>Tf-Idf</i> , <i>n-gram count</i> .
Agrawal and Amol (2016) [27]	A convolution neural network based model is proposed to detect clickbaits.	<i>Click-Word2Vec</i> , <i>Click-scratch</i>	CNN	Created their own corpus from Reddit, Facebook, and Twitter social media platforms	<i>Click-scratch</i> - 89% accuracy with 0.87 AUC-ROC score; <i>Click-Word2Vec</i> -90% accuracy with 0.90 AUC-ROC score

Table 4.10 continued from previous page

Chakraborty <i>et al.</i> (2016) [4]	Build a browser extension to automatically detect the clickbait headlines.	Sentence Structure, Clickbait Language, Word patterns and n-gram features	SVM, Decision Tree, Random Forest	Collected 30,000 headlines (clickbait and non-clickbait) from ViralStories, Upworthy, BuzzFeed, Wikinews, Scoopwhoop, and ViralNova.15,000 headlines both in the clickbait and non-clickbait categories	SVM-93% accuracy rate with 0.95 precision, 0.90 recall, 0.93 F1-score and 0.97 AUC-ROC values; Decision Tree-90% accuracy rate with 0.91 precision, 0.89 recall, 0.90 F1-score and 0.90 AUC-ROC values; Random Forest-92% accuracy rate, 0.94 precision, 0.91 recall, 0.92 F1-score and 0.97 AUC-ROC values using a combination of all extracted features.
--	--	---	-----------------------------------	---	--

Table 4.10 continued from previous page

Khater <i>et al.</i> (2018) [157]	A supervised machine learning algorithm was applied to classify the posts into clickbait and non-clickbait headlines.	Extracted 28 features. The most common are Bag of Words (BOW), noun extraction, similarity, readability, and formality.	Logistic regression, Linear SVM	The dataset used was provided by Bauhaus-Universitat Weimar at the time of clickbait detection challenge.	Logistic regression and Linear SVM gave 0.79, 0.78 precision and 0.79, 0.79 recall values, respectively.
-----------------------------------	---	---	---------------------------------	---	--

Table 4.10 continued from previous page

Our proposed approach	A two-phase hybrid CNN-LSTM Biterm model is proposed to detect short topics (clickbait headlines) and its types.	Without pre-trained vectors, with <i>Word2Vec</i> and <i>GloVe</i> (100 dimensional) pre-trained vectors	CNN + LSTM	First dataset [4] consists of 32,000 (clickbait and non-clickbait) headlines. Second dataset [157] consists of 12,000 (clickbait and non-clickbait) headlines. Our created corpus (both textual and non-textual) from Reddit and Facebook social media platforms consists of 1,800 (clickbait and non-clickbait) headlines.	The accuracy achieved by our proposed hybrid CNN-LSTM model for Dataset 1, Dataset 2 and Dataset 3 using <i>GloVe</i> embeddings is 95.8%, 89.44% and 94.21%, respectively. Whereas the accuracy achieved by the proposed hybrid CNN-LSTM model using <i>Word2Vec</i> embeddings for Dataset1, Dataset 2 and Dataset 3 is 93.42, 84.87 and 89.96, respectively.
-----------------------	--	--	------------	---	---

It was observed that our proposed hybrid CNN-LSTM model with *Word2Vec* embeddings outperforms [27] using *click-Word2Vec* variant in terms of accuracy, precision and F1-score rates by 0.96%, 3.97% and 2.70%, respectively.

López-Sánchez *et al.* (2017) used metric and deep learning algorithms by integrating them with CBR (Case-Based Reasoning) methodology using *n-gram*, *Word2Vec*, *Tf-Idf* feature extraction techniques [182]. The proposed model achieved 0.99, 0.95, 0.90 average area under ROC curves using *Word2Vec*, *Tf-Idf*, *n-gram count* on *Dataset 1* [4]. Whereas our proposed approach achieves 0.99 AUC-ROC score using *GloVe* embeddings for the same dataset by precisely identifying the type of clickbait headlines that are spread on social media. Automatic identification of classified clickbait headlines is not seen in the case of [182]. Khater *et al.* (2018) used the dataset created by Bauhaus-Universitat Weimar [157] giving 0.79, 0.78 precision, 0.79, 0.79 recall and 0.79, 0.79 F1-score values using Logistic regression, and Linear SVM models. Whereas, the proposed model gives 0.95 precision, 0.89 recall, 0.92 F1-score values using *GloVe* pre-trained vectors on the same dataset. Chakraborty *et al.* (2016) build a browser extension to automatically detect the clickbait headlines using *sentence structure*, *clickbait language*, *word patterns*, and *n-gram* features [4]. Our proposed hybrid CNN-LSTM model using *GloVe* embeddings outperforms SVM, Decision Tree, Random Forest in [4] by 2.8%, 5.7%, 3.8% accuracy rates and 0.02, 0.09 and 0.02 AUC-ROC scores, respectively.

4.7 Discussions

Interesting characteristic differences between clickbait and non-clickbait headlines have been highlighted in this chapter. Such characteristics are used as

features to classify clickbait headlines. A two-phase model has been proposed. In the proposed approach, the hybrid CNN-LSTM model is implemented in the first phase to identify clickbaits, which are further fed to the second phase where the BTM is implemented to analyse the category of clickbait headlines. The proposed approach outperforms when compared to the existing clickbait systems. It was observed that the *Shocking/Unbelievable*, *Hypothesis/Guess* and *Reaction* types of clickbait headlines are published maximum on social media platforms to attract the readers. To the best of our knowledge, no work has been done to detect the type of clickbait on subsequent visits by the users of social media.

Some of the limitations of our proposed system is that, only the images with English textual data can be considered for detecting clickbait headlines. Images without textual content cannot be classified by the proposed system. Also, the proposed system is trained on the collected dataset, so can give biased results when unrelated headlines are predicted. Extraction of textual content from images without using any language check or grammar error library yields poor performance. Statistical significance for the comparison of the results retrieved from the existing system with the proposed model in terms of p-values, confidence interval are not supported. Also, the proposed method uses different feature extraction techniques so the qualitative comparison with existing systems was not possible.

In future, the end user will be able to identify the type of clickbait headline through a browser extension. The proposed system has the potential to provide an impulse to analyse the type of clickbait spreading at the time of political elections, natural disasters, terrorist attacks, *etc.* It will become easier for the researchers to classify the category of clickbait spread at maximum during natural events. Such type of application can help in controlling

the various types of clickbaits from growing over social media by activating a pop-up message indicating the warning in form of clickbait categories (*Number, Questionable, Reasoning, Reaction, Revealing, Shocking/Unbelievable, Hypothesis/Guess, Forward referencing*).

Chapter 5

DeepFakes: Temporal Sequential Analysis to Detect Face-Swapped Video Clips using C-LSTM

Social media is flushing up with deepfake (using face-swapping tools) videos of users. Deepfake videos are AI-generated clips that use open source libraries like Google image search, Tensorflow [183], social media [164, 61] websites (like YouTube [10] videos, stock photos, and Instagram [184]) to insert famous people faces onto similar pre-existing background videos by creating a machine-learning based algorithm [111].

In 2017, an anonymous user of Reddit posted defamatory videos of multiple celebrities in compromising positions [26]. Though the videos were not real but damaged the identity/reputation of various world leaders, famous celebrities, politicians. Deepfakes exists in three different forms [18] as follows.

- Lip-sync - In this form, a source video is modified to make an arbitrary

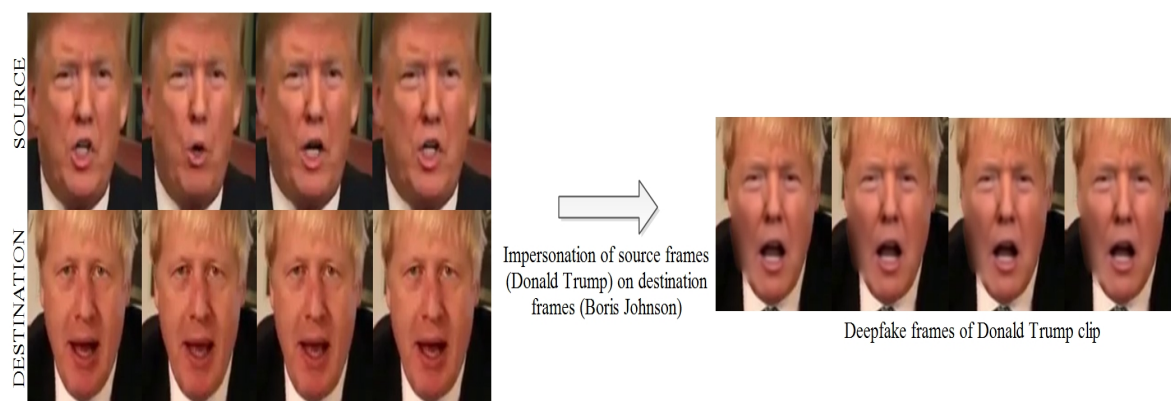


Figure (5.1) Impersonation of source frames retrieved from source clip onto destination clip to extract the frames for generating deepfake video clip

audio recording showing a consistent movement along the mouth region. For instance, the director and actor Jordan Peele used this technique to create a viral video of Obama saying inflammatory things about president Trump.

- Puppet-master - It is a type of technique where a target person (puppet) is animated using their facial expressions, eye movements and head movements by an actor. The actor performs by acting out in front of the camera to show the puppet's actions.
- Face-swap - Two videos (source and destination) are considered in this technique. A face in the destination video is replaced by the source face. For example, one could create a deepfake of Donald Trump (US president) by superimposing its face onto a video of Boris Johnson (UK president) by doing an impersonation of Trump as shown in Figure 5.1.

Among these (lip-sync, Puppet-master, face-swap), the most common one is

the face-swap form of deepfake (also known as AI-synthesized media) [114, 110]. In this chapter, we will focus on face-swap deepfake news clips related to famous politicians. Also, the deepfake technology acts to be a double edged-sword [1] by playing a creative role for the advancement in virtual reality, production and film editing.

Software tools are designed to allow people, without a technical background or programming experience, to create deepfakes. With enough photos and video content of the target, usually sourced through publicly available social media profiles, hiring an expert to develop fake videos of unsuspecting victims is not difficult. Multiple frames pulled from one or more videos can help gather the few hundred images to develop fake videos [108].

Deepfakes can have a devastating impact on their victims, leading to traumatic stress disorders, depression, anxiety and in extreme cases, driving them to the point of suicide. It can smear the victim's public reputation, resulting in unemployment, layoffs, *etc.* The technology is relatively easy to use, especially with readily available machine learning algorithms and open-source codes. The process also requires just a basic consumer-grade graphics card to get the job done within hours.

In this chapter, we created deepfake videos (100 source and 100 destination) dataset related to famous politicians using GAN-based approach is presented. It is an open-source [185] developed from the original autoencoder-based *Deepfake algorithm* [186] for face-swaps. Based on the collected dataset, 100 deepfake video clips of reputed politicians are generated. Further, deepfake face-swapped video [26] clips of politicians are detected using temporal sequential frames retrieved from the clips to protect the political leaders against deepfakes within a timespan of initial 2 seconds of uploaded video clip. The proposed approach uses the forged video [113] to extract the frames

at first level followed by which a deep depth-based Convolutional Long Short-Term Memory (C-LSTM) model is used to identify the fake frames at second level.

5.1 Motivation

The spread of deepfake videos where faces of politicians are swapped when they deliver speech during election campaign cannot be examined by the general public to analyse the video clip as deepfake or real. It is analysed that with the upcoming 2020 US elections [10] will be influenced by deepfake contents spread over social media platforms through shares, retweets, *etc.*

Many deepfake clips [43] on the Internet are featured with fake porn clips of female celebrities for earning money through social media platforms. The first deepfake featuring Indian celebrities was published by Pornhub user ‘blackandwhitepanda’. The person has posted two videos related to Indian celebrities. The videos come with a deepfake disclaimer, disturbing the society as it is made up of non-consensual content. A deepfake video of ‘Wonder Woman’ was went viral on the social media when the actress was seen in a compromising position. Some Reddit user used a machine learning algorithm, an easily accessible software and open-source code to morph ‘Wonder Woman’ face on the girl in the video clip.

The spread of such contents (images, audios, videos) on social media platforms serves a approach for generating false information by targeting politicians, celebrities leaving a negative impact on the society. As deepfakes have become easier to produce and is cheap to create, so apart from politicians, companies across geographies, industries are also at a greater risk. So, there is an urge to detect and prevent such contents from spreading. Most

of the existing techniques do not make assumptions on distribution of user behaviour for contextual content of data and the models developed do not score the validity of the deepfake video, audio clips spread over social media platforms. Hence, it becomes difficult to classify the real or fake content at real-time. Existing systems do not apply deep learning models for the verification of domain name, where the actual content is published. The existing systems do not put emphasis for detecting the degree to which content is invalid, fake or not retrieved from the source. The existing real-time systems to detect fake content are relied on images, posts, but no such system exists which can detect fake video content. Also, a comparison study depicts that the existing systems to detect fake content do not give accurate results for analysing real-time deepfake contents.

So, there is an urge to detect manipulated videos as deepfakes seen in the form of fooling people into thinking they are receiving instructions from a trusted individual.

5.2 Contributions

The key contributions of this chapter are as follows.

- A ground truth dataset (deepfake and real frames) has been prepared by swapping famous politicians face to the pre-existing video clips.
- A novel approach is proposed which works under a two-level structure.
- At first level, the forged frames from deepfake video is extracted using *OpenCL* and in next phase pre-processing is performed on the extracted frames to feed it to the next level. At second level, an existing deep temporal-based Convolutional Long Short-Term Memory (C-LSTM)

model is used to identify the fake frames to detect the fake face-swap video clips.

- Prediction of the deepfake video clips on the basis of temporal sequences and inconsistencies between the retrieved frames from the LSTM layer of C-LSTM model to build a highly efficient model.
- The proposed deep temporal-based C-LSTM model is compared with the state-of-the-art models on the basis of performance metrics and training time.

5.3 Problem statement

The proposed work addresses an issue to identify fake news videos of famous politicians. To solve the listed problem, we propose a novel approach to analyse the deepfake features in the tampered videos.

The videos are classified using a binary $\{deepfake_video, real_video\}$ classifier and the classification process is performed at two levels. At first level, real and deepfake video clips are selected. Further, frames are extracted from the referenced video (real and deepfake) clips. Following which face alignment is performed on the extracted frames to perform pre-processing. At second level, a deep temporal-based Convolutional Long Short-Term Memory (C-LSTM) model is applied to identify the inconsistencies between the extracted frames to predict deepfake video clips.

5.4 Deepfake videos exposed

As the *DeepFaceLab* generates the deepfake videos, the temporal and the intra-frame inconsistencies between frames are created [186]. To detect the deepfake manipulation, the temporal and intra-frame anomalies can be exploited. Let us briefly explain the way we can exploit such video anomalies and the way a deepfake video is generated to detect the deepfake video clips.

5.4.1 Training and generation of video using DeepFace-Lab

Neural networks (autoencoders) are used to compress or decompress the images [187]. Figure 5.2 shows the chosen image (retrieved face) in case of face-swap is fed to an encoder which gives a low dimensional representation of face which is also known as a latent face. The latent face is then passed to the decoder for its reconstruction. To make face-swapping possible, the *DeepFaceLab* uses two autoencoders with the same encoder and different decoders [110].

For training a neural network, two sets (source and destination) of training images are required. Among these two sets, the first set consists of original frames which will be replaced by the destination video frames resulting in the creation of a manipulated destination clip. The second set consists of the faces that will be swapped with the source video frames [187].

The two sets (source and destination clips) are treated separately. The Decoder of X (Donald Trump) is only trained with the retrieved frames of its clip (source); the Decoder Y (Boris Johnson) is only trained with the frames retrieved from its clip (destination) as shown in Figure 5.2. Whereas, all the common features are identified by the encoder from the latent faces [188].

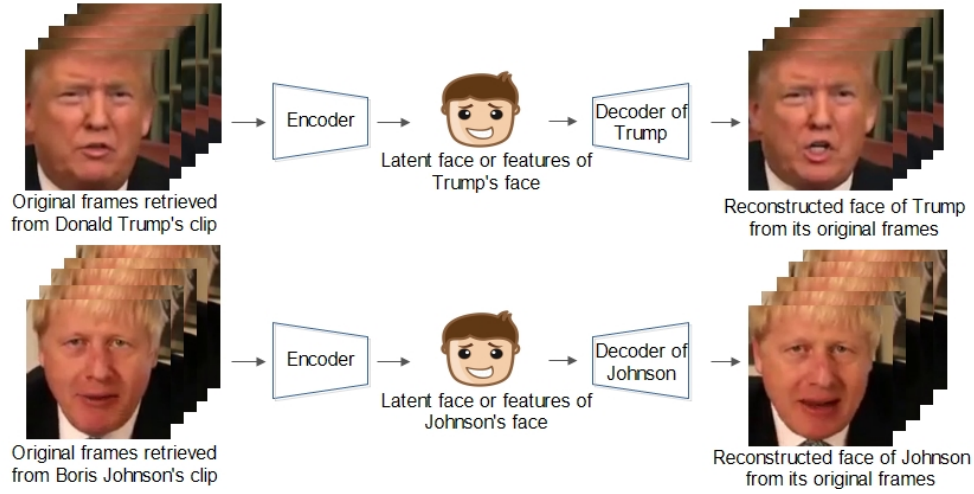


Figure (5.2) Representation of two networks (encoder and decoder) for the source and destination clips during the training phase

Automatically the encoder can identify common features in both faces itself as these faces share a similar structure [189].

After completion of the training process, a latent representation of a face generated from the source clip X (Donald Trump) present in the source video is passed to the decoder of Y (Boris Johnson) to reconstruct subject X from Y as shown in Figure 5.3.

Also, the intermediate iterations extracted during merging phase to generate deepfake video clips is shown in Figure 5.4.

Further, a swapped video [112, 190] will be retrieved which will be used as a first frame-level feature to be taken as an input for our proposed deep temporal-based C-LSTM model. To perform the frame-level feature extraction, a face detector will be used to extract the face regions from the whole frame as shown in Figure 5.5 which will be further passed to the trained autoencoder to generate deepfake video clips [185].

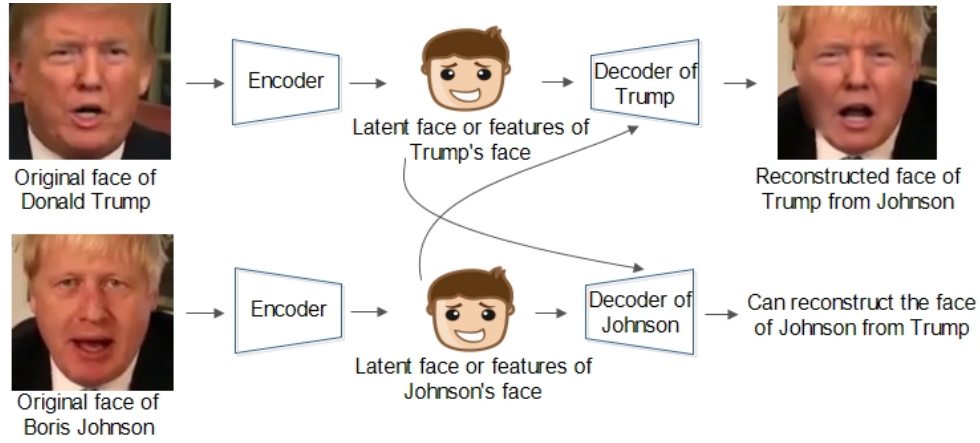


Figure (5.3) Representation of two networks (encoder and decoder) for the source and destination clips during the generation phase

5.5 Deepfake tools

Several advanced DeepFake existing tools can be used to create tampered video clips as depicted in Table 5.1.

5.6 Data Collection

Many sources such as Twitter [12], LinkedIn [61], Facebook [75], YouTube [10] are the common trading platforms to publish inappropriate information. Such type of information are disseminated in the form of images, videos, posts on social media websites [61].

5.6.1 Experimental Setup

To perform the experiment, deepfake dataset has been collected by replacing a politician's face from a source video and imposing it onto the destination video using a Neural Network (NN) [190, 191]. The minimum requirements



Figure (5.4) Merging of (a) Arvind Kejriwal frames of source clip onto Ashok Chavan frames of destination clip at iteration 60, 113570 and 234050, (b) Donald Trump frames of source clip onto Boris Johnson frames of destination clip at iteration 60980, 197301 and 138254, and (c) Jaya Bachchan frames of source clip onto Smriti Irani frames of destination clip at iteration 37594, 97927 and 156121 using DeepFaceLab

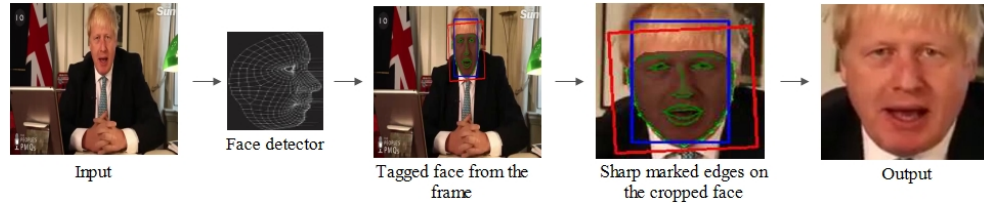


Figure (5.5) Extraction of the face region from the retrieved frames of the video clip using face detector

Table (5.1) Summary of existing deepfake tools

Tools	Models	Key features	Source
Faceswap	Encoder-Decoder method	Uses two sets of encoder-decoder, allows to share the parameters to other clips	https://github.com/deepfakes/faceswap
Faceswap-GAN	Generative Adversarial networks	Two new features (VGG and adversarial loss) are added to the Faceswap tool, its an upgraded version of Faceswap-GAN. A new mask for the face is added to the model which eliminates noise and produces better skin tone.	https://github.com/shaoanlu/faceswap-GAN
DeepFaceLab	Neural network (Autoencoders)	Supports multiple face attraction features, uses different model as per compatibility of the system. Uses H-64 simple PC, SAE for simple graphic card PC, Saehd for advance graphic PC.	https://github.com/iperov/DeepFaceLab
DFaker	Convolutional neural network	Detects specific face landmarks. The advanced loss function is used to recreate the face.	https://github.com/dfaker/df
DFaker-tf	Convolutional neural network	It is similar to Dfaker, but having high-quality results. It removes noise from the background of the frames. MSE is used from the background of the frames. Inputs and outputs are fixed, i.e., 64×64 and 128×128 images, respectively.	https://github.com/StromWine/DeepFake_tf

Table (5.2) Summary of configurations required for the collection of dataset

Developer	Url	Configuration used
iperov	https://github.com/iperov/DeepFaceLab	16Gb RAM NVIDIA video card with 8GB video Windows 7 and higher OpenCL-compliant graphics card (NVIDIA) Processor supporting SSE instructions
System	Item	Details
GPU	Titan Xp	CUDA cores : 3840 Graphic clock : 1404 MHz Memory data rate : 11410 MHz Memory Interface : 384-bit Memory Bandwidth : 547.68 GB/s Bus : PCI Express x16 Gen3 Dedicated video memory : 12288 MB GDDR 5x

to generate deepfake video clips is discussed in Table 5.2.

To generate a deepfake video, a *DeepFaceLab* application is required to be downloaded from the GitHub repository [186]. Further, cloning of the repository by building each Python library is required. The software required to run the application is CUDA toolkit *9.0* with all the four patches (CuBLAS update with RNN and Convolutional, CuBLAS update of GEMV operation, CuBLAS update with new GEMM kernels, and CuBLAS update with GEMM heuristics) along with the base installer.

The other software required to run the application is Nvidia CuDNN *7.0* for CUDA *9.0* toolkit. The step-wise instructions to execute the repository using the above discussed software is given in [192]. With the help of the *DeepFaceLab* application, deepfake videos are generated using source and destination video clips (a sample of 20 videos from our collected dataset).

To perform experimentation a large dataset (in the form of video frames) is created, which consists of 200 videos (100 real and 100 deepfake) for evaluating our proposed model. The whole dataset has been curated from

Table (5.3) A sample of 20 Person of Interest (POI) chosen for source and destination video clips from our collected dataset with high-quality downloads

Name	Persons of Interest (POI)		Video Duration (minutes) + Quality (HD)	
	Source interest	Destination interest	Source clip	Destination clip
SD_1	Donald Trump	Borris Johnson	01:15 + 360	01:05 + 360
SD_2	Arvind Kejriwal	Ashok Chavan	00:34 + 360	00:38 + 360
SD_3	Jaya Bachchan	Smriti Irani	00:35 + 720	00:30 + 360
SD_4	Rahul Gandhi	Narendra Modi	01:15 + 360	00:23 + 360
SD_5	Arvind Kejriwal	L.K. Advani	00:31 + 360	00:30 + 720
SD_6	Akhilesh Yadav	Arvind Kejriwal	00:49 + 360	00:47 + 360
SD_7	Mahua Moitra	Hema Malini	00:32 + 360	00:37 + 360
SD_8	Smriti Irani	Sasikala Pushpa	00:58 + 480	00:46 + 480
SD_9	Hina Rabbani Khar	Mahua Moitra	00:41 + 360	00:38 + 480
SD_{10}	Narendra Modi	Hukmdev Narayan Yadav	00:30 + 480	00:32 + 480

YouTube channel related to famous politician’s news or speeches [9]. Two types of video clips (source and destination) are considered for our dataset, where the face of the politician used in the dataset in a source video is swapped with the face of the politician in the destination video. Focus was to concentrate on the videos of a Person of Interest (POI) talking in formal scenarios such as public speech, news interview and weekly address.

All videos were manually downloaded where the primary focus was on the POI facing towards the camera. The proposed approach is tested on the famous politicians as shown in Table 5.3.

The SD_i represents the face-swap between source and destination clips, where $i \in \{1, 2, \dots, 100\}$, S represents the source video and D represents the destination video. After merging the source over destination video, the deepfake video is named as SD . In total 200 video clips (source + destination) are collected to perform the experiment. The source video clips are

Table (5.4) A sample of 20 Person of Interest (POI) video clips from our collected dataset with total time taken to train the model, total loss after training the model, total number of frames retrieved from these clips.

Name	Loss-value	Training duration (hrs)	Number of frames (at 2 sec)	Total frames
SD_1	0.01	10:08:34	62	1962
SD_2	0.11	15:32:08	60	1143
SD_3	0.07	19:00:58	62	924
SD_4	0.02	16:23:41	64	717
SD_5	0.07	12:30:48	26	373
SD_6	0.01	21:07:38	54	1261
SD_7	0.04	14:55:12	36	668
SD_8	0.06	13:28:05	36	792
SD_9	0.07	12:47:53	50	923
SD_{10}	0.02	15:14:11	34	541

swapped with the destination by overlapping the facial features from source to destination using *DeepFaceLab* application where the background of the destination video remains unaltered. It has been observed from Table 5.4 that the number of frames retrieved for 2 second video clip is highest for SD_4 . Also, the time-based representation (in millisecond) per frame retrieved from the deepfake clip of Donald Trump is shown in Figure 5.6.

The three frame sequence of a fake clip from the original clip using *DeepFaceLab* from our collected dataset is shown in Figure 5.7. Where X (Arvind Kejriwal), Y (Donald Trump) and Z (Jaya Bachchan) (from the collected source clips), P (Ashok Chavan), Q (Boris Johnson) and R (Smriti Irani) (from the collected destination clips) are swapped and returns deepfake frames of source clips by impersonating it onto the destination frames.

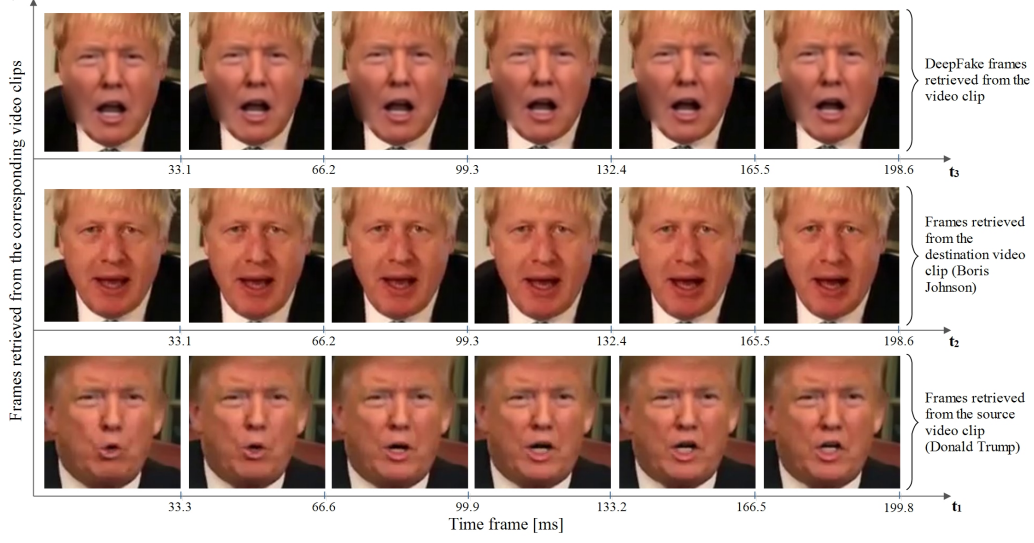


Figure (5.6) Time-based representation of video frames retrieved from the source, destination and deepfake video clips.

5.7 Methodology

An overview of the proposed automatic detection system for deepfake clips is represented by Figure 5.8. A set of unseen test sequences is passed to the CNN model to retrieve the features for each sequential frame generated during video training and generation as discussed in the previous section. The extracted features are then passed as an input in the form of sequential frames to the LSTM model. At last, an estimation of the likelihood of the sequential frames being real or deepfake is detected.

The detailed architecture of the proposed C-LSTM model for automatic detection of deepfake clips are shown in Figure 5.9. The system is composed of two levels, *i.e.*, Image pre-processing at first level, Convolutional Neural Network [193] and LSTM subnetworks [194] at second level for processing sequential frames of both video clips (source and destination).



Figure (5.7) Three deepfake example frame sets generated from the source video clips using DeepFaceLab

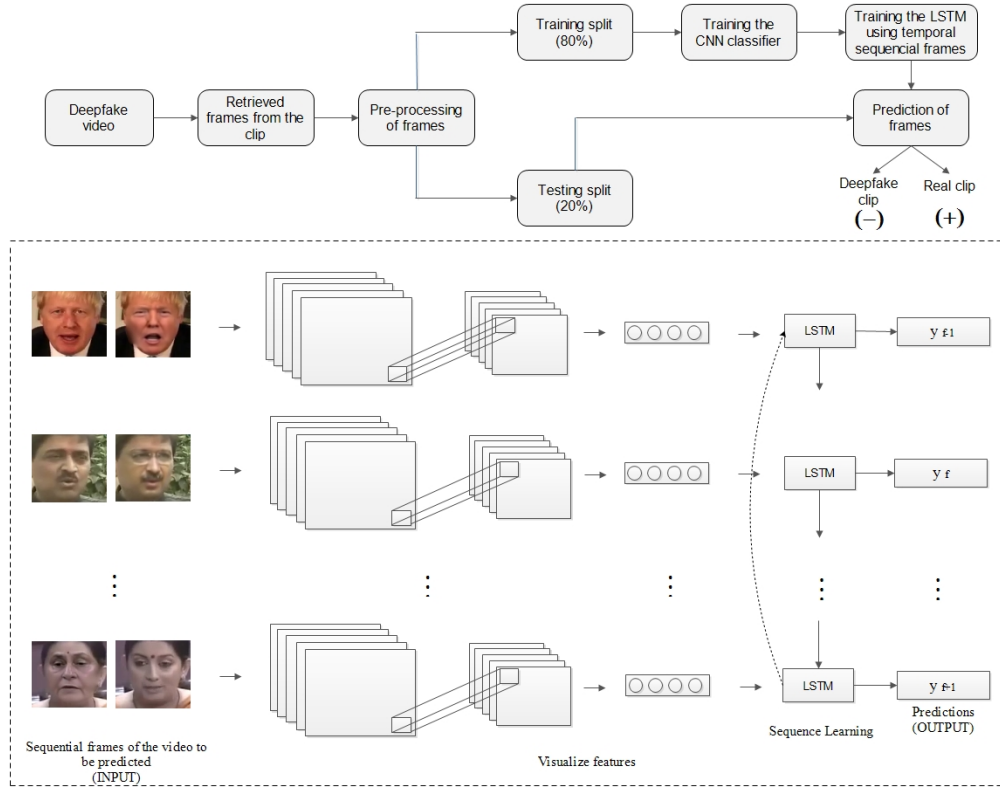


Figure (5.8) Overview of our proposed deep temporal-based C-LSTM model for automatic detection of deepfake video clips of politicians. It consists of Convolutional-Long Short Term Memory for processing the input temporal frames.



Figure (5.9) Two-level structure of the proposed deep temporal-based C-LSTM model for automatic detection of deepfake face-swapped video clips of politicians

5.7.1 Image pre-processing

The input frames retrieved from the video clip use *ImageDataGenerator* class to perform image pre-processing. Various operations performed for image pre-processing phase as seen in level 1 in Figure 5.9 are discussed below.

- Image rescaling: The original frames in the form of images consist of RGB coefficients in the range $(0-255)$. These values are too high to feed directly into the proposed model, so the values are re-scaled between 0 and 1 using the $1/255$ scaling factor.
- Shear mapping: It is performed on the set of frames where each image is displaced from its edge to the vertical direction. The *shear_range* parameter controls the displacement rate and the deviation angle between the horizontal line of the original frame and the frame of a line in the transformed frame (*shear_range=0.2*).
- Zooming augmentation: To make the appearance of the face in the frame more larger, zooming augmentation is configured by the *zoom_range* (*0.2*) parameter. The range of this parameter varies from $[1 - value, 1 + value]$, i.e., 0.2 zoom value will have the range from $[0.8, 1.2]$.
- Horizontal flip: The zoomed images are then flipped horizontally by setting the boolean value of the *horizontal_flip* parameter to 'True'.

The two other essential components of the proposed system are CNN and LSTM as discussed below.

5.7.2 Convolutional-Long Short Term Memory (C-LSTM)

An end to end learning of fully connected layers are used to detect the deepfake clip of politicians as shown at level 2 in Figure 5.9. Our proposed model can be divided into CNN and LSTM components [195]. CNN is used to extract the high-level features from the sequential frames of the source and destination video clips. LSTM is used for capturing the inconsistencies, temporal-based sequences and reduces the training time of the model. With the help of LSTM, it becomes easy to analyse the temporal sequences of the video frames to improve the efficiency of the model, as an automatic detection of a deepfake video clip is done within a timespan of initial 2 seconds of input video clip.

The training and testing frames include source, destination and deepfake generated video clip frames. Where, source and the destination video clip frames are real and the generated video clips using DeepFaceLab tool are fake. The input sequence of frames retrieved for 80% training data after pre-processing is then fed to a C-LSTM [196]. The extracted features of frames are detected at different regions using sliding filter vector evolved in the convolution layer.

Consider $u_i \in \mathbb{R}^m$ as the m -dimensional vector of a particular frame from the input video clip for i^{th} position. Let $u \in \mathbb{R}^{l \times m}$ be the input frame sequence, where l denotes the length of the sequential frame. Consider p as the length of the filter and vector $v \in \mathbb{R}^{p \times n}$ be the filter for performing convolution operation. A window vector w_j with p consecutive frame vectors, where j is the position of a pixel in a frame is given by Eq. (5.1).

$$w_j = [u_j, u_{j+1}, \dots, u_{j+p-1}] \quad (5.1)$$

Here, commas represents the row vector concatenation. The vector v revolves at each position with the window vectors to generate a feature map $fm \in \mathbb{R}^{l-p+1}$, where each element fm_i of feature map for window vector w_j is represented by Eq. (5.2).

$$fm_j = n_t(w_j \odot v + b) \quad (5.2)$$

Here, $\odot$ is the element-wise multiplication, n_t is the non-linear transformation function and b is the biased term that belongs to $\mathbb{R}$.

The number of filters used in the CNN model for our experiment is 32 of 3×3 dimension, *i.e.*, the convolution window is set to 3. *ReLU* is the non-linear activation function chosen for the hidden layers and 2×2 is the pool size for the max-pooling layer which remains same for all three max-pooling layers in CNN model [193].

During the learning process, the proposed model is trained for 25 epochs. The input shape of the frame considered in the architecture is 128×128 . The frames are split into 80:20 (training:testing) ratio. The input dimension (128×128) is fed to the first convolutional layer (a). The output dimension of the first convolutional layer (a) is $126 \times 126 \times 32$, which is fed as an input to the first max pooling layer (b) giving an output of $63 \times 63 \times 32$. The second convolution layer (c) takes $63 \times 63 \times 32$ dimensions as the input and trains the vector. In next step, the second max-pooling layer (d) takes $61 \times 61 \times 32$ input shape from the previous convolutional layer (c) and gives $30 \times 30 \times 32$ as the output shape. Further, the third convolutional layer (e) uses 64 feature maps and gives $28 \times 28 \times 64$ as the output shape which is fed to the third max-pooling layer (f) as shown in Figure 5.9.

After the third max-pooling layer (f), flattening is performed. The 2048 - dimensional feature vectors retrieved as an output from the flattening layer

are used as the sequential input to the LSTM (g) with 256 neurons through input gate (i_g) as shown in Eq. (5.3).

$$i_g = \sigma(W_j \cdot [h_{t-1}, x_t] + b_i) \quad (5.3)$$

Where h_{t-1} is the previous output of the hidden state, x_t is the input at current time step t and σ is the logistic sigmoid function that gives output between $[0,1]$.

The sequence of CNN feature vectors is recursively processed in LSTM (g) to address the temporal sequence of video frames with 16384 (values) $\times$ 256 (neurons) + 256 (bias values for neurons) as output. LSTM is followed by a dense layer (h) with a 0.5 chance of dropout which is capable to remember the temporal sequence of input frames using *ReLU* function. In the dropout layer (h), randomly selected neurons are ignored during training which helps to make the network less sensitive to the specific weights of neurons. Hence, the network becomes less likely to overfit the training data. A second dense layer (with *Sigmoid* function) (i) is used to give the final prediction of the proposed network classifying a sequence being deepfake or real video. The number of frames to be extracted from the video clip is decided by the end-user. If the aggregation of the score for fake video is greater than 50% then it is decided that it is a deepfake video clip else it will be real video clip.

Hence, the CNN model is used to extract the high-level features from the selected video clips and LSTM model is used for capturing the inconsistencies seen with the extracted frames. Various results have been evaluated and discussed in next section to test the proposed model.

5.8 Evaluating C-LSTM model

The proposed model was implemented using Theano [179]. Theano is a Python library that supports the efficient use of a GPU and symbolic differentiation. The model was trained on a GPU (Titan Xp) system to get better efficiency.

To evaluate the proposed deep temporal-based C-LSTM model, a comparison of performance analysis using MesoNet [125], Capsule [197], CNN state-of-the-art models have been discussed in this section. Classification of deepfake video clip [198] has been performed on the collected dataset using the proposed model.

The analysis of training and validation data on the basis of accuracy metric, and loss information for collected deepfake video dataset using state-of-the-art models and our proposed model at different epochs is shown in Figure 5.10. It has been observed from Figure 5.10b, Figure 5.10d, Figure 5.10f, that the MesonNet [125], Capsule [197] and CNN model gives a validation loss of *1.81*, *46.93*, *6.31*, respectively, which is greater than the proposed model calculating the validation loss of *1.75* as shown in Figure 5.10h. Similarly, the CNN model gives a validation accuracy of *97.06%* as observed from Figure 5.10e, which is less than the proposed model having the validation accuracy of *98.21%* as shown in Figure 5.10g, thus affecting the model efficiency.

The comparative analysis between the proposed and other state-of-the-art models on the basis of validation accuracy and loss history at every epoch level is shown in Figure 5.11. The comparative analysis of precision metric is shown in Table 5.5.

Consider the number of epochs (*epoch*)_{*i*} where $i \in \{1, 2, \dots, 25\}$. It has

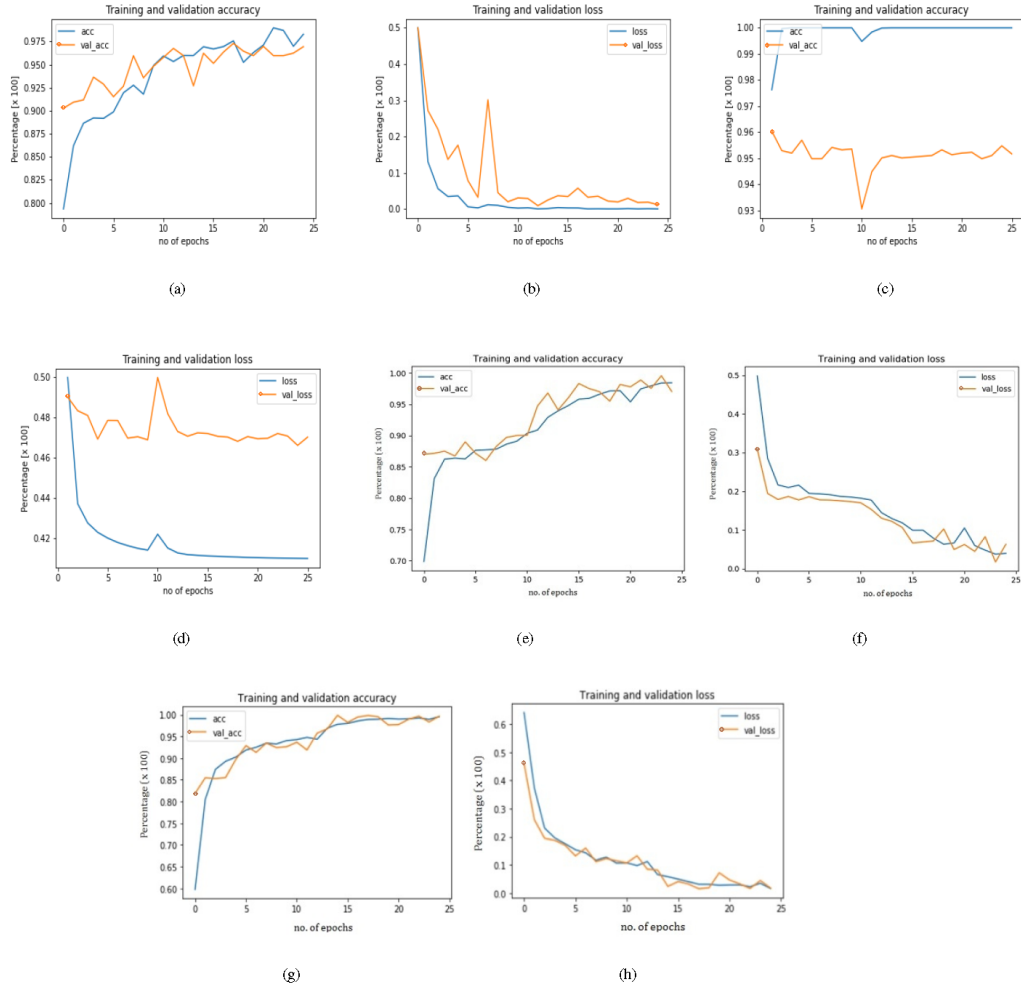


Figure (5.10) Analysis of (a) Training vs validation accuracy and (b) Training vs validation loss for collected deepfake video dataset using the MesoNet model, (c) Training vs validation accuracy and (d) Training vs validation loss for collected deepfake video dataset using the Capsule model, (e) Training vs validation accuracy and (f) Training vs validation loss for collected deepfake video dataset using the CNN model, (g) Training vs validation accuracy and (h) Training vs validation loss for collected deepfake video dataset using the proposed deep temporal-based C-LSTM model at different epochs

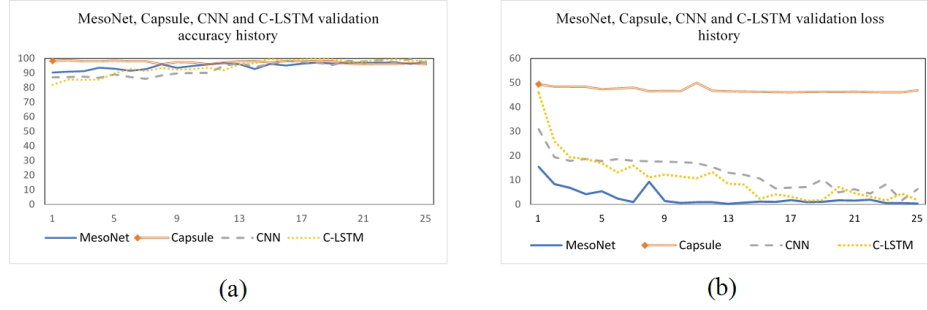


Figure (5.11) Comparison of state-of-the-art vs proposed deep temporal-based C-LSTM (a) validation accuracy history, (b) validation loss history at different epochs

Table (5.5) Comparative analysis of performance measures using MesoNet, Capsule, CNN and the proposed deep temporal-based C-LSTM architecture on our collected dataset

Architecture	Frames		Accuracy	Precision	Recall	F1-score	Loss	Training Time
	Fake frames	Real Frames						
MesoNet	90,804	90,804	97.30	98.91	93.97	96.37	1.81	00:58:43 hr
Capsule			96.03	97.99	93.29	95.58	46.93	03:49:23 hr
CNN			97.06	97.91	94.24	96.05	6.31	01:22:23 hr
Temporal-based C-LSTM			98.21	99.62	93.91	96.68	1.75	00:39:42 hr

been observed from Figure 5.11 that the C-LSTM model accuracy metric $<$ CNN model accuracy metric for $(epoch)_i$ where $i \in \{1, 2, 3, 4\}$. A similar behavior of validation loss history at each epoch level has been analysed. It was observed that C-LSTM model has the validation loss $>$ CNN validation loss at $(epoch)_i$ where $i \in \{1, 2, 3\}$ and gradually the loss rate decreases till 1.75 for C-LSTM at $(epoch)_{25}$. Hence, the proposed C-LSTM model is better than the simple CNN based on the accuracy and value-loss information metrics.

The performance of the C-LSTM was evaluated in terms of accuracy, precision, recall, F1-score as shown in Table 5.5. The results of the performance metrics for all state-of-the-art and the proposed temporal-based C-LSTM models have been shown in the same table as discussed above. It has been analysed that the proposed model outperforms MesoNet [125], Capsule [197] and CNN state-of-the-art models when LSTM is used for detecting deepfake clips on the basis of temporal sequences and inconsistencies seen between retrieved frames from our own collected dataset.

The total number of real and deepfake frames retrieved from the collected dataset is 18,1608. Our proposed model achieves an accuracy of 98.21% for 0.9962 precision and 0.9391 recall values. The loss value of validation data is 1.75, which is quite lesser than the CNN (6.31), Capsule (46.93) validation loss. Also, the total training time taken by the MesoNet (00:58:43 hr), Capsule (03:49:23 hr), CNN model (01:22:23 hr) $>$ C-LSTM model (00:39:42 hr) which is collectively greater than the existing state-of-the-art models. Through these observations, it was analysed that the proposed model improves the overall performance with 1.15% in comparison to the simple CNN model. The proposed model reduces the total training time to train the features of the deepfake video frames.

Table (5.6) Comparative analysis of performance measures using Celeb-DF, DFDC and ground dataset using proposed deep temporal-based C-LSTM architecture

Proposed temporal-based C-LSTM model							
Datasets	Fake frames	Real frames	Accuracy	Precision	Recall	F1-score	Training Time
Celeb-DF	1,23,900	1,23,900	96.46	98.99	98.99	98.99	00:56:91 hr
DFDC	1,87,900	1,87,900	96.99	99.07	97.15	98.10	01:35:26 hr
Ground truth	90,804	90,804	98.21	99.62	93.91	96.68	00:39:42 hr

No standard dataset is available for detecting deepfake video clips of politicians. So, mixed domain dataset (Celeb-DF [114] and DFDC [48]) available online and observed that our proposed deep temporal-based C-LSTM model gives effective results in terms of F1-score and accuracy metric as shown in Table 5.6. Also, from Table 5.5, it was analysed that our proposed model outperforms the state-of-the-art techniques that were used to detect the deepfakes on our collected ground truth dataset.

5.9 Major findings

Some of the important findings that can be observed after evaluating the proposed model are:

- It was really hard to run the proposed model on Celeb-DF and DFDC publicly available datasets due to out of memory issues.
- The total training time taken to train the model on Celeb-DF and DFDC from scratch is more than the generated ground truth dataset.
- The proposed temporal-based C-LSTM model outperforms other state-of-the-art models in terms of accuracy, F1-score, loss.

- The training time taken by the proposed model is less than the MesoNet, Capsule, CNN models when trained on the generated ground dataset.
- The structure of the proposed temporal-based C-LSTM model is easy to understand and allows to extract the basic features from the retrieved frames. The model uses only three hidden layers which helps to reduce the training time of the proposed model.

5.10 Discussions

A two-level temporal-based C-LSTM model has been proposed. In the proposed approach, the forged frames from deepfake videos are extracted using *OpenCL* and a frame-level pre-processing is performed at the first level. The pre-processed frames are further fed to a C-LSTM algorithm to analyse the deepfake (face-swapped) videos. The deepfake clips are predicted on the basis of temporal sequences and inconsistencies between the frames retrieved from the LSTM layer of C-LSTM model which achieves better performance than other state-of-the-art models. Also, it was observed that if a POI in the clip consistently looks away from the camera, the deepfake video detection is compromised.

The dataset created with the help of human annotations is restricted to only politicians news, speech related video clips consisting of famous politicians. During video collection, various key constraints are required to be followed like a person in a clip should be facing towards camera to capture the facial frames properly, the background of the video should not be merged with the facial features of the POI. The generated dataset is restricted to only second generation collection of deepfake video clips and the proposed model performs best for only second generation dataset. Third generation dataset

is not considered for the proposed approach. To classify the percentage of frames extracted as real or fake, a statistical proven method for extracting threshold value is not used.

In future, the detection of deepfake can be done using the edited audio by changing the audio speech of a person along with lip-sync. The source clip frames can be further extended to include the images from publicly available sources to analyse more realistic results. The statistical analysis of the system can also be considered to retrieve p-value or confidence interval to compare the optimized model for detecting deepfake video clips.

Chapter 6

A Cross-Platform Application to avoid Fake Web Trap

In this chapter, the developed *IsItFake* Progressive Web Application (PWA) and its features are discussed. It promotes the exchange of information among end-users. The purpose of creating such web application is to expedite the users to detect different types of fake or unverified content seen in the form of quotes, videos, articles, clickbaits over social media platforms. The goal of developing such an application is to outreach the social media users to give the power to analyse the reliable, trustworthy and real content for social development.

6.1 Contributions

As the spread of such fake contents creates a huge massive impact on society. The fake news articles, quotes in form of images, clickbait headlines, deepfake video clips are spreading at a faster pace and human-eye cannot

observe the variation between real and fake contents [164, 14, 110]. So, it is important to build an automated real-time fake content detection system. A web-based progressive application (*IsItFake*¹) has been developed to provide an integrated and novel solution together to detect fake news articles, quotes in form of images, clickbaits, deepfake video clips over social media platforms to reduce the risks posed by the spread of such contents. Our contribution to this application is as follows.

- A post-processing approach was proposed on the extracted text to analyse the non-verified image quotes.
 - Extraction of real author names from the misquoted images.
- An ideal multi-level voting classifier has been proposed to classify the positive and negative instances of the articles.
 - Extraction of textual content from the URL to extract necessary features to identify fake news articles.
- A two-phase structure, using CNN-LSTM hybrid model at first phase and Biterm Topic Model (BTM) at second phase has been proposed to identify clickbait headlines.
 - Extraction of textual content from the URL (clickbait links) to identify the categories of clickbait headline.
 - Graphical analysis has been performed to classify the type of clickbait headlines using BTM.
- A two-level novel approach has been proposed to identify deepfake video clips. At first level, pre-processing is performed and at second level, a

¹<https://isitfake.co.in/>

deep depth-based C-LSTM model is used to identify the fake frames of deepfake video clips.

- User-friendly analysis performed on the retrieved frames on the basis of time interval.
- Statistical analysis of multimodal signals (text, video, images, URLs) using Python web framework-Django.
- Designed a generic approach for data-science related cross-platform application for end-users.
- Public consumption of a RESTful API to identify fake or unverified contents at real-time.
- A web-based application is developed to expedite the users to detect different types of fake content seen in form of quotes, news articles, clickbaits and deepfake video clips.

6.2 Methodology

IsItFake cross-platform is embedded with four applications. The overview of various phases formed in all the four applications (quote examiner, fake news article detector, clickbait detector, deepfake video detector) is shown in Figure 6.1.

The quote examiner model is proposed to verify quotes prior to its usage in order to differentiate a fake quote and retrieve the real author of the quote. An OCR tool and Google Cloud Vision are used in the proposed model for text detection [199]. To enhance the accuracy during retrieval task from the quoted images, a post-processing approach has been applied where automatic

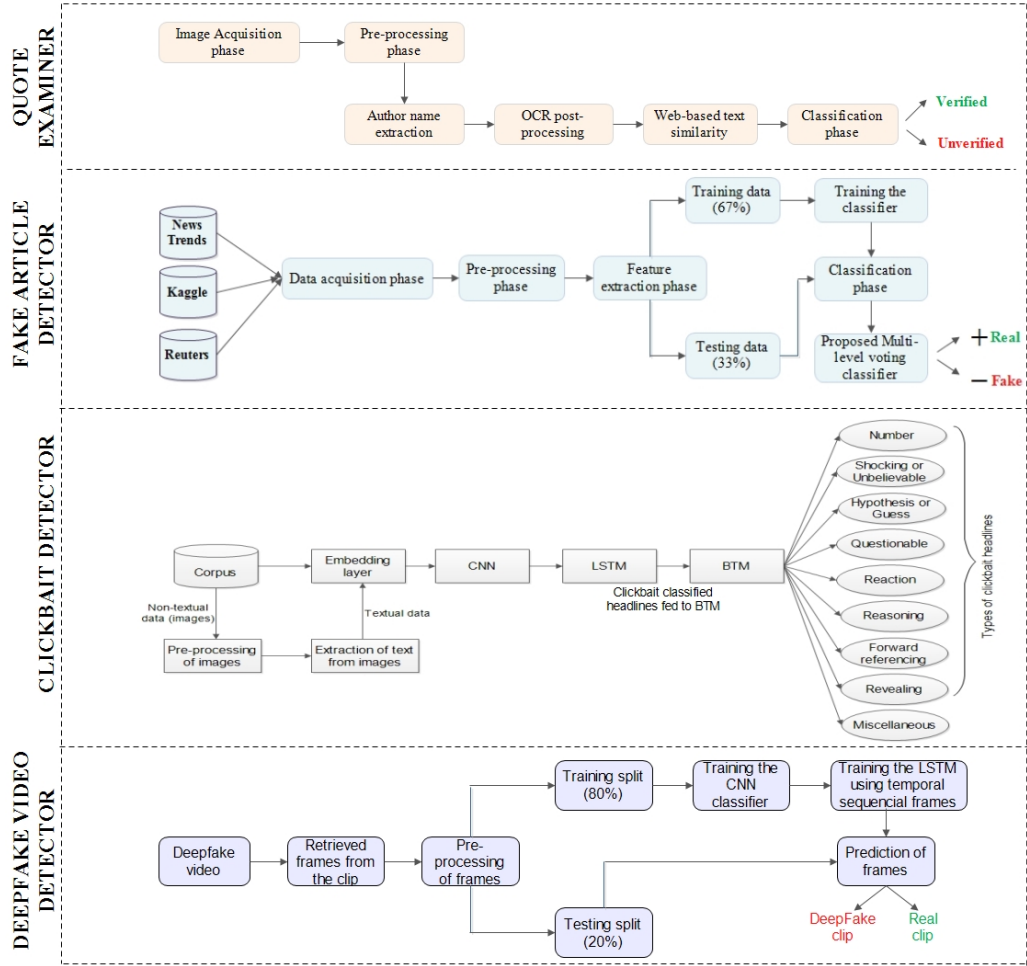


Figure (6.1) Overview of proposed models used in *Isitfake* cross-platform

removal of spelling errors are corrected. Further, to validate the source of the textual data retrieval, a Google search engine is used to perform a web-based similarity search for textual data [14].

The flow diagram of the proposed fake news article detection system [164] is seen in Figure 6.1. The extracted text from the collected dataset is fed to the pre-processing phase. To retrieve the meaningful features from textual content, feature extraction phase is carried out. The retrieved features are then fed to the proposed multi-level voting model in next phase to identify real or fake news articles. The detailed architecture of the proposed system is discussed in Chapter 3.

The methodology followed for the proposed two-phase clickbait detection model is shown in Figure 6.1, where the hybrid Long-Short Term Memory (LSTM) model and Convolutional Neural Network (CNN) [196] is embedded at the first phase. At second phase, BTM [14] is used to identify the eight types of online clickbait headlines over social media platforms. The detailed architecture of the proposed system is discussed in Chapter 4.

An overview of the proposed deepfake detection of face-swapped video clips of politicians is seen in Figure 6.1. A set of unseen test sequences is passed to the CNN model to retrieve the features for each sequential frame generated during video training and generation of deepfake content in form of videos [110]. The extracted features are then passed as an input in the form of sequential frames to the LSTM model. At last, an estimation of the likelihood of the sequential frames being real (untampered) or deepfake (tampered) is detected. The detailed architecture of the proposed system is discussed in Chapter 5.

Various sources (Facebook [75], Twitter [10], YouTube, Reddit [26]), which disseminate fake contents were used to collect the dataset to train

the proposed models. All the four proposed models use different datasets, pre-processing approach, feature extraction techniques, machine learning and deep learning models as shown in Table 6.1.

6.3 Preliminaries

The *IsItFake* system runs on any mobile device containing a Chrome or Edge browser. To qualify *IsItFake* as a Progressive Web Application (PWA) [201], it needs to have a secure connection (HTTPS), responsiveness, work offline with service workers, contains a standard manifest file specifying experience and behaviour of the application. PWAs do not need to be installed like traditional applications nor are required to be uploaded on Play Store or App Store for users to download. It can be readily installed through opening the website link directly and selecting ‘Add to Homescreen’ dialog.

The features of the developed PWA system (*IsItFake*) are discussed in next section.

6.3.1 Features of the proposed system

The problem to detect verified quotes, fake articles, clickbait headlines, face-swapped videos can be seen as a two-stage classification process where an input sample has to be assigned in one of two classes: verified/real or unverified/fake. Further the process in second stage will permit to find a set of discriminant features to retrieve the probability score of the unverified/fake content through the progressive web application.

The developed PWA (*IsItFake*) will load data on devices having low Internet connectivity [201]. The developed system will be able to handle advance web features like push notifications that are not available with most other

Table (6.1) Analysis of the proposed systems

Proposed system	Preprocessing techniques	Feature extraction techniques	Models	Datasets
A quote examiner model is proposed to verify the image quotes. The classified fake quotes are further processed to retrieve the real author of the quote.	Converting image to gray scale and removing noise from images, spell-check and grammar-check	URL extraction of similar text are retrieved from images with the help of Google search engine	Non-local Means Denoising algorithm	Kaggle Handwritten Word (2017), KAIST scene text (2012) and IIIT 5K-word (2012) [200]
Multi-level voting model is proposed which helps to reduce the training time and improve the accuracy of the classifiers.	All the missing values, duplicate textual content and stop words are removed from news articles	Hashing-Vectorizer, Count-Vectorizer and Tf-Idf.	MultinomialNB, SGD, PA, CART, LinearSVC, NuSVC, SVC, LR, MLP, AdaBoost, Gradient Boosting, Voting)	News Trends (2017), Kaggle (2017), Reuters (2004)
A novel two-phase approach for modelling short topic contents in form of clickbait headlines has been proposed.	Removal of extra whitespaces, stop-words. OCR tool is used to process non-textual data to to extract the textual content and Non-local Means Denoising is used to remove noise from images.	Word2Vec and GloVe	Non-local Means Denoising algorithm, CNN-LSTM, BTM	Chakraborty (2015), Khater (2017), Created own dataset (2019)
An C-LSTM model is proposed automatic detection system for automatic detection of deepfake video clips of face-swapped politicians using temporal sequential frames within a timespan of 2 seconds.	Image rescaling, shear mapping, zooming augmentation and horizontal flip	NA	CNN and LSTM	Celeb-DF (2019), DFDC (2019), Created own dataset (2020)

applications. To improve the security issues, all the data traffic will pass through HTTPS connection with a TLS 3.0 security, which is a latest and most advance handshake mechanism. To ensure responsiveness of the website, an open source CSS Framework based on Flexbox, known as, *Bulma* is used. Its modular components helps in designing and prototyping applications at a rapid pace. The entire platform is handled by one framework - Django which is a Python based web framework, used in designing clean and pragmatic web applications.

6.4 System architecture

The proposed system structure is shown in Figure 6.2. The landing page of our developed platform is shown in Figure 6.3 of *IsItFake* ² application. The page contains four buttons which provides services for quote examination, fake article detection, clickbait detection, deepfake video clip detection. Server used in the system has been developed using Django framework. Client side design has been developed using *Bulma* CSS framework.

²<https://isitfake.co.in/>

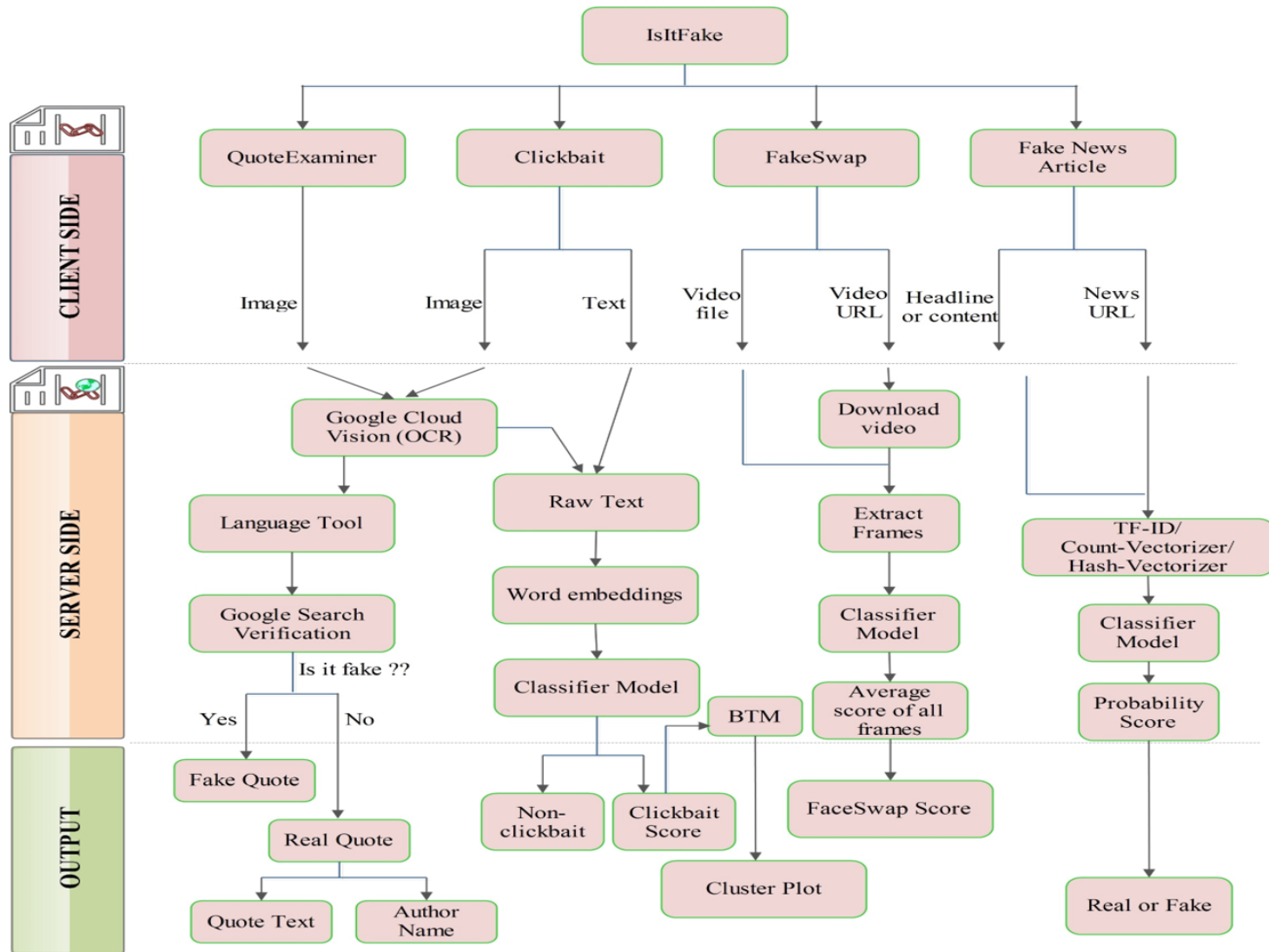


Figure (6.2) System architecture

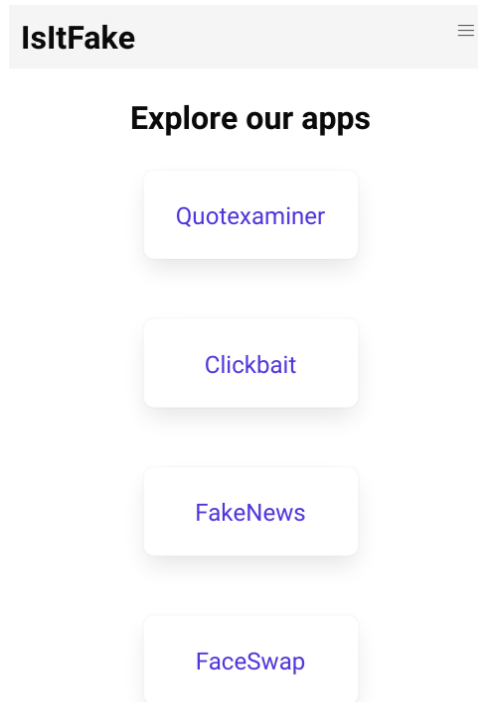


Figure (6.3) Landing page of *IsItFake* platform

6.4.1 QuoteExaminer

In QuoteExaminer service, image file (.png, .jpeg) is fed to Google OCR to extract the text from it. After text extraction, the cleaning process on the language using language checker module is performed [14]. Text check Python libraries like *language_check* and *autocorrect* are used to correct the non-word and real-word errors to ensure that the retrieved text is grammatically correct. Further, the text is passed to the Google Search API to get relevant links. A total of 15 top links are targeted from the result. Each link is tested for authenticity. If the links are authentic and relevant to the quote, server fetches the author of the quote and sends back the response.

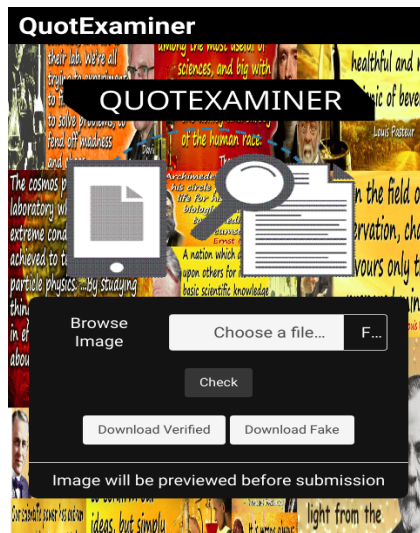
Various screenshots of the QuoteExaminer system are shown in Figure 6.4. Dashboard of the proposed system is represented by Figure 6.4a. The input preview of the verified and unverified quote is given by Figure 6.4b and Figure 6.4c, whereas the output preview of the verified and unverified quote is represented by Figure 6.4d and 6.4e.

Client side

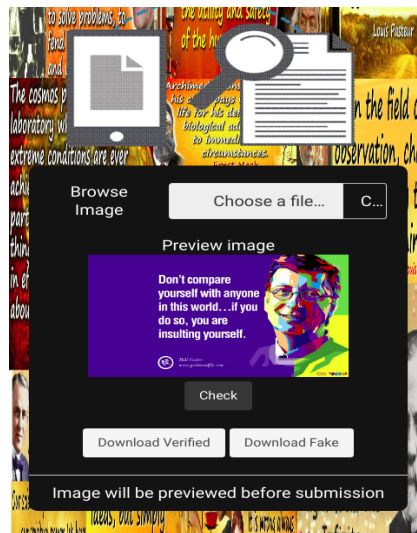
QuoteExaminer is used for providing the authenticity of the quote and their corresponding authors. In the QuoteExaminer service, users should submit an image file containing with supported extensions - *png, jpeg*.

Server side

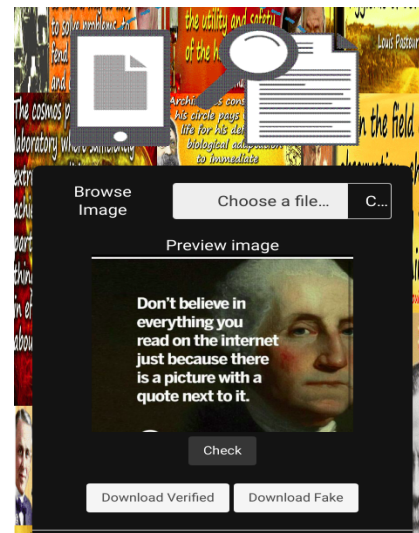
In QuoteExaminer service, image file is fed to Google OCR to extract the text from it. Further, the retrieved text is cleaned by using language checker module. The text is passed to Google Search API to get relevant links. If the links are authentic and relevant to the quote, server fetches the author of the quote and sends back the response.



(a) Dashboard



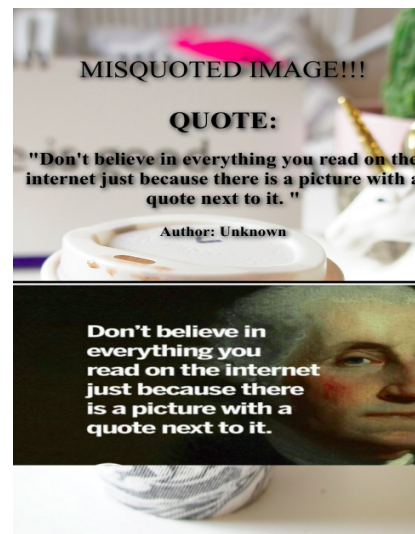
(b) Preview of verified quote



(c) Preview of unverified quote



(d) Output of verified quote



(e) Output of unverified quoted

Figure (6.4) Screenshots of QuoteExaminer system

6.4.2 Fake news article

FakeNews service includes support to parse a news URL as well as raw text. The extracted content is directly fed to the loaded model object for prediction [164]. An alert message is given as an output to the end-user on the basis of final model prediction. Dashboard of the Fake news article detection system is shown in Figure 6.5a. User can enter the content in form of textual information or can copy the URL directly to extract the textual content from the website as represented by Figure 6.5b. After clicking on the submit button, the model gives the predicted output as shown in Figure 6.5c and 6.5d.

Client side

Fake news article system is used in classifying news content into fake or real. The system service accepts news URL or raw news content to predict whether it is fake or real.

Server side

FakeNews service includes support to parse a news url as well as raw text. For extraction of headline and content from URL, *Newspaper3k* module is used to scrape the content data. Pickle library is used to load the model and vector transformation files. The extracted content is directly fed to the loaded model object for prediction. Output from model contains whether the content is real or fake. User is alerted based on the model prediction.

Fake Article Detector

Verify article from content or raw Url

Content

e.g. news content here

OR

URL Link

e.g. <https://www.nytimes.co>

Submit Reset

(a) Dashboard

Fake Article Detector

Verify article from content or raw Url

Content

Here is an image of <http://cbsnews.com.co/> taken on March 17, 2017 from the Internet Archive WayBack

OR

URL Link

e.g. <https://www.nytimes.co>

Submit Reset

(b) Form with content preview

Fake Article Detector

Output

Predicted:
real

Input content:
Earlier this week, the country surpassed the case totals in China and Italy. The number of known cases has risen rapidly in recent days, as testing ramped up after weeks of widespread shortages and delays. Over 600,000 cases have now been confirmed worldwide, according to

(c) Output of real news article

Fake Article Detector

Output

Predicted:
fake

Input content:
Saudi Health authorities have announced a possible breakthrough in the development of a treatment for the Covid-19, saying that people already infected with the well-known Camel Herpes virus could be immune to the pandemic. The Arab kingdom has reported barely 1100

(d) Output of fake news article

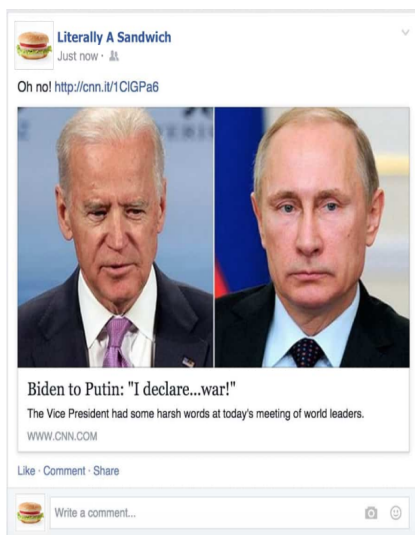
Figure (6.5) Screenshots of Fake news article detection system

6.4.3 Clickbait Detector

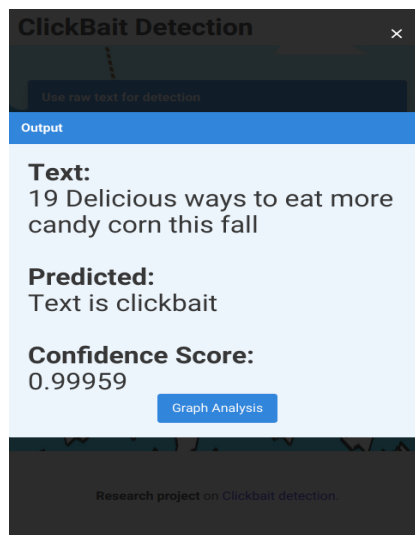
Clickbait detection service accepts raw text or an image file (.png, .jpeg). If the user submits an image, it goes through Google OCR for textual content extraction [14]. The maximum sequence length to feed in a model is 100. If the proposed model gives positive prediction as an output, *i.e.*, predicts the textual headline as a clickbait then a BTM model is used to define the cluster class. Corpus data is converted into 9 different classes such as *hypothesis*, *shocking*, *reaction*, *questionable*, *reasoning*, *forward referencing*, *revealing*, *number* and *miscellaneous*. The detected clickbait headline with confidence score is retrieved as shown in Figure 6.6b. It is then fed to BTM to analyse the type of cluster as shown in Figure 6.6c. Similarly the clickbait detector system also takes the input in form of images as shown in Figure 6.6d.

Client side

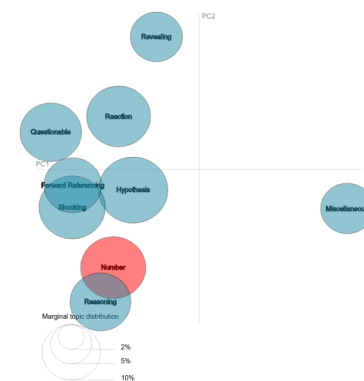
Clickbait helps in providing assurance whether a given textual headline is misleading or not. Every service consists of an input form with fields to upload media content. In the clickbait service, users can upload an image file or raw text content. Once submitted, a model will pop up with output containing class of prediction, probability score and a button to view the graphical analysis only if the system predicts the textual headline as clickbait. The analysis part shows a cluster map with a highlighted cluster as represented by Figure 6.6c.



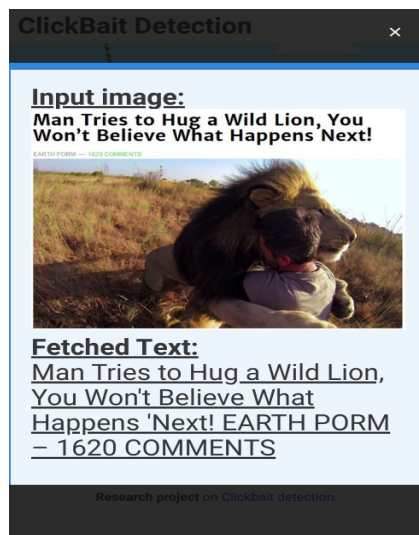
(a) Dashboard



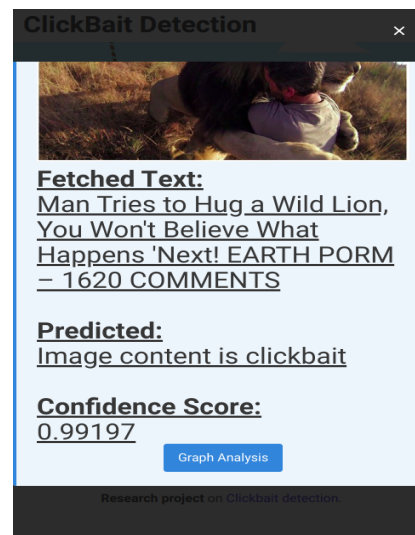
(b) Output of clickbait headline



(c) Clickbait cluster analysis



(d) Preview of clickbait image



(e) Output of confidence score

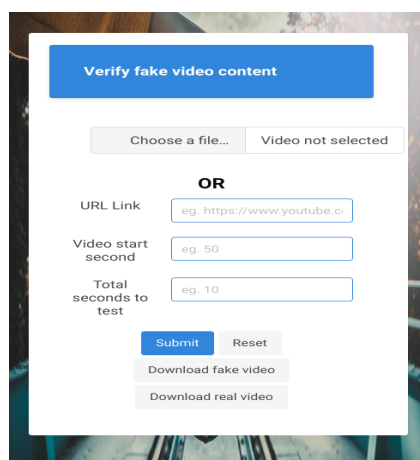
Figure (6.6) Screenshots of Clickbait detector system

Server side

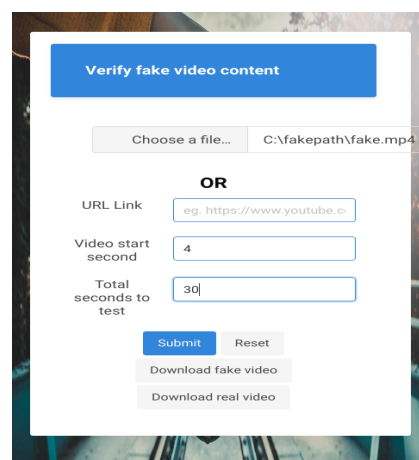
Clickbait detection application accepts raw text or an image file. If the user submits an image, it goes through Google OCR for textual content extraction. Pre-processing part includes Keras preprocessor modules - *Tokenizer* and *pad_sequences*. *Tokenizer* is used to convert the text into numeric tokenized array. The maximum vocab size is 20000 tokens. Pad sequence is used to append 0 in the sequence. The maximum sequence length is 100. After pre-processing the array is fed to the prediction model. For loading the model on server, Keras *load_model* is used. Based on the prediction output, BTM model is used to define the cluster class. Corpus data was converted into 9 different classes. Class names are *hypothesis*, *shocking*, *reaction*, *questionable*, *reasoning*, *forward referencing*, *revealing*, *number* and *miscellaneous*.

6.4.4 Deepfake swap

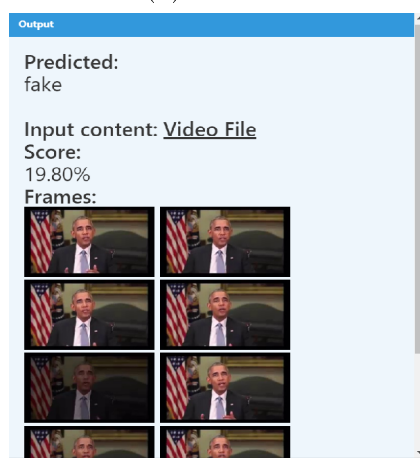
Deepfake swap service supports raw video content file or a video URL. By default the frames are extracted for the first, middle and the last second of timespan of the video clip by default, but the user has the option to enter the starting second of the clip and the total number of frames to extract from the video as shown in Figure 6.7b. Every extracted frames are saved onto the server and displayed to the end-user as represented by Figure 6.7c. The extracted frames are passed onto the loaded model for classification of frame. Output of each frame is averaged to compute the final prediction as given in Figure 6.7d. Various screenshots of the proposed system are shown in Figure 6.7.



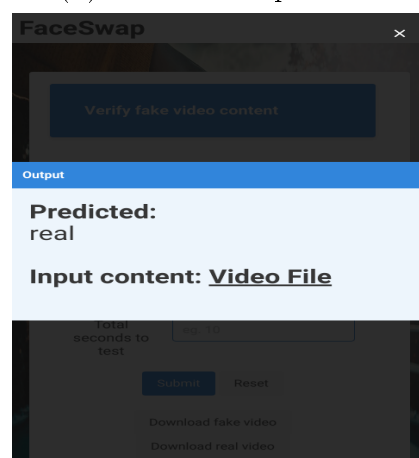
(a) Dashboard



(b) Form with input content



(c) Output of fake video



(d) Output of real video

Figure (6.7) Screenshots of deepfake FaceSwap detection system

Client side

Deepfake swap detection system helps in analysing video frames and classifying the clipped video into fake or real. The system allows users to upload a video URL or a direct raw video content file. End-user can also mention the start frame second and the total number of seconds for frame extraction of the consecutive seconds. This feature is a optional one.

Server side

Deepfake swap service supports raw video content file or a video URL. Uploaded content is parsed accordingly. For a video URL, it is first being downloaded onto the server for processing. Video content is downloaded using modules *PyTube*, *urllib* and *requests* module based on the url. All these modules helps in downloading all types of video URLs. Once a video file is available, server will extract frames from it for every second. By default the frames are extracted for the first, middle and the last second of the video. Also, the user has the option to send the start second of the frame and total frames to extract from that point. Every extracted frame is saved onto the server. For video analysis, *OpenCV* module is used for frame extraction and meta data computation for any uploaded video. For pre-processing of each frames, Keras image module is used to load the frame and converting it into vector form. Tensorflow object is used to load the trained model file. Every vector array is passed onto the loaded model for classification of frame. Output of each frame is averaged to compute the final prediction. The threshold for real video is 0.5. Any output more than 0.5 is termed as a fake video.

6.5 Implementation

On client side HTML, CSS and JavaScript is used and on server side Django framework is used to develop the RESTful APIs. Nginx is used to act as reverse proxy and serve as a load balancer. For saving data in DB, PostgreSQL Database is used which is widely supported in a Django application. In production hosting, Gunicorn server is used to spawn multiple Django applications, which will help to serve large number of requests from Nginx server. The trained model files are loaded into the memory on startup of each application. During less traffic time, the application goes in sleep mode to ensure that the server resources are not utilised during no traffic time. The number of applications to spawn depends on the traffic load and the server resources availability. To serve user request with 200 status code, uptime guarantee of server and rightful use of resources should be used.

At the client side, user uploads the inputs like images or videos or links as per the service inside a form. These inputs are passed to the server using HTTP POST multipart request. In the request part, media file contents are appended using multipart feature of HTTP requests. This helps to pass different set of data in the same request, *i.e.*, plain text and media file. Based on the request object, server processes the inputs and sends back the response object containing the results. A single framework is used to handle the front-end and back-end, server supports data transmission in form of JSON or HTML parsing based on the request client. For serving third party users who want to consume our API, Django REST Framework module of Django is used to create APIs for public. This APIs return JSON if user explicitly mentions to return a JSON object. Further, this response can be used in any front-end frameworks. By default the response contains HTML

object to be rendered on the browser.

For converting a Django application into a PWA, *django-pwa* module is used. It auto generates the required files like *manifest.json* and *service-worker.js* for a PWA. Manifest file contains information about the application. Service worker file helps in caching the latest updates of the application for offline usage. Along with cache file which consists of a manifest file, *i.e.*, the configuration file of any PWA, it contains application name, start URL, and the logos with different dimensions to support all kinds of devices which are browser compliant. Manifest file handles all the meta data of the application.

6.6 Discussions

Isitfake is a Progressive Web Application developed to benefit the end-users by identifying the fake contents seen in the form of quotes, articles, clickbaits and videos over social media platforms.

The proposed system is only restricted to image, video and textual contents. QuoteExaminer system depicts the verified or unverified quotes on the basis of the collected database. The predicted results for retrieving author names remain unknown when the extracted content from the quoted images do not match with the database. The application does not works for the deepfake audios and non-English fake textual contents. For scrapping URLs to detect fake news articles, restricted websites are considered.

In future, the developed application will be able to give the pop-up to the end-users to get an intimation of fake content at real-time by pointing cursor to the quotes, images, videos, blogs, articles and clickbaits available over social media platforms. Also, the graph analysis will be displayed for

the identified fake words through the pop-up messages. With the help of these analysis graph, the end-users will be able to visualize the probability of fake words involved among textual content in disseminating fake data over Internet sources. Also, a tool will be made publicly accessible (comprising of front-end and back-end) for detecting a video clip as tampered or original at real-time. Its back-end will consist of a pre-trained model (deep temporal-based C-LSTM) and the front-end will consist of browser plug-in (Google Chrome) which will communicate over HTTP RESTful APIs. Such application can help in controlling the various types of fraud videos (related to political speeches) from growing over social media platforms.

Chapter 7

Conclusion and Future scope

This thesis focuses on efficiently identifying the fake contents, which are seen in form of various multimodal signals (images, videos, headlines, articles). This chapter concludes the research work presented in this thesis and presents the future scope of the work.

7.1 Conclusions

In this thesis, a detailed description is provided to detect fake contents seen in form of multimodal signals over social media platforms. The generic process including dataset collection, pre-processing, feature extraction, fake content detection for both textual and non-textual data, output analysis phases is discussed in detail. A comprehensive literature review for fake content detection over social media platforms for different multimodal signals (images, articles, quotes, clickbaits, deepfake videos) has been performed. A proper methodology was followed to perform the detailed review. The review was performed by identifying the related research studies from the renowned elec-

tronic databases as well as the top most conferences related to the area. To further narrow down the count of selected studies, inclusion-exclusion criteria was followed. Further, the extracted outcomes are arranged on the basis of their year-wise publications. Different sources for workshops, conferences, journals were used to download the research articles. Also, various online freely datasets available were discussed. Various types of existing fake content detection techniques using machine learning and deep learning models are also discussed in detail. The year-wise increase in research area to detect fake content online is represented by graph to help the researchers to know the latest research work.

Three approaches has been proposed in this thesis work. To detect fake news articles, a multi-level voting model has been proposed. Various machine learning models are trained to build the voting classifiers using standard ground datasets. Three feature extraction techniques (Tf-Idf, CV, HV) are used to extract the vector representation from the textual content of the articles. The methodology along with its components followed by the architectural framework of the proposed model to detect the fake news articles has been visualized. The proposed multi-level voting model not only helps to improve the accuracy but also helps to reduce the training time of the classifiers. Discussions are done related to the reduction seen in training time of the proposed model. The testing benchmark and results of the proposed multi-level voting model with various state-of-the-art models has been compared.

The second approach proposed for detecting clickbait headlines efficiently identifies the textual content from the images using the OCR tool and perform post-processing technique to improve the accuracy of the retrieved text. A web-based text similarity is used to obtain the URLs of similar text ob-

tained from the images and verify the extracted domain names against the authentic quotation websites database. The overall methodology and framework of the proposed two-phase hybrid CNN-LSTM structure to detect the clickbait headlines from textual and non-textual dataset (collected and own created) using *Glove* embeddings has been explained in depth. The embedded BTM with CNN-LSTM is used for automatic identification of the eight types of clickbait headlines into *reasoning*, *number*, *reaction*, *revealing*, *shocking/unbelievable*, *hypothesis/guess*, *questionable*, *forward referencing*. At last, cluster analysis to detect the quality of nine clusters and comparative study is performed with the existing systems to show the effectiveness of the proposed model.

Third approach has been proposed to identify deepfake video clips. A ground benchmark dataset has been created for deepfake video clips of famous politicians. DeepFaceLab tool is used to generate deepfake video clips. A two-level C-LSTM structure is proposed to detect the deepfake video clips of politicians efficiently. At first level, the forged frames from the video clip is extracted and further pre-processing is performed on the extracted frames to feed it to the second level. A temporal-based C-LSTM model is used to identify the fake face-swap video clips. The framework to detect deepfake video clips and the architecture of the proposed system is discussed in detail. The proposed model is tested on our collected dataset on the basis of performance metrics and training time and is compared with various state-of-the-art models .

An integrated Progressive Web Application (PWA) is developed to benefit the society. The system is integrated with four proposed models to identify fake contents seen in form of quoted images, news articles, clickbait headlines and deepfake video clips.

7.2 Future Work

The proposed systems can be used to identify the fake contents at real-time over social media platforms. In future, the traditional techniques like Tf-Idf, CV and HV for extracting features from the news articles can be replaced by word embeddings like *GloVe*, *Word2Vec* for making the fake news article detection system more independent. The proposed multi-level voting model can be trained on other multilingual content for detecting fake news articles. For clickbait detection system, long textual content with more than 100 terms can be considered to predict the headlines in future. Video-based dataset can be created for extracting clickbait headlines. For deepfake video detection, third generation dataset can be used for improving the efficiency of the proposed C-LSTM model. The deepfake dataset created with the help of human annotations is restricted to politicians news, so other fields related to celebrities, natural disasters can be gathered in future.

The developed PWA can be used to give an alert message as a pop-up on screen for the fake contents seen on the current web page. Also, the GUI of the developed system can be made more interactive, so that the end-users can analyse their predicted results more easily.

In future, it will become easy for the researchers to classify the category of clickbait, fake news article, deepfake video spread during natural events. Such type of application can help in controlling the various types of fraud videos (related to political speeches), the spread of fake contents during elections, terrorism, natural calamities, crimes from growing over social media platforms for the betterment of the society. Also, our future plan is to study individual fake content sub-problems and build the in-depth integrated solution for fake content detection.

List of Publications

- Sawinder Kaur, Parteek Kumar, Ponnurangam Kumaraguru, “Automating Fake News Detection System using Multi-level Voting Model”, Soft Computing, (Impact Factor - 2.78)
- Sawinder Kaur, Parteek Kumar, Ponnurangam Kumaraguru, “Detecting Clickbaits using Two-layer Hybrid CNN-LSTM Biterm Model”, Expert systems with Applications, (Impact Factor - 4.57)
- Sawinder Kaur, Parteek Kumar, Ponnurangam Kumaraguru, “Deep-fakes: temporal sequential analysis to detect face-swapped video clips using convolutional long short-term memory”, Journal of Electronic Imaging, (Impact Factor - 0.92)

References

- [1] H. Ahmed, I. Traore, and S. Saad, “Detection of online fake news using n-gram analysis and machine learning techniques,” in *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, pp. 127–138, Springer, 2017.
- [2] P. Dewan, M. Gupta, K. Goyal, and P. Kumaraguru, “Multiosn: real-time monitoring of real world events on multiple online social media,” in *Proceedings of the 5th IBM Collaborative Academia Research Exchange Workshop*, pp. 1–4, 2013.
- [3] W. Wei and X. Wan, “Learning to identify ambiguous and misleading news headlines,” *arXiv preprint arXiv:1705.06031*, 2017.
- [4] A. Chakraborty, B. Paranjape, S. Kakarla, and N. Ganguly, “Stop clickbait: Detecting and preventing clickbaits in online news media,” in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 9–16, IEEE, 2016.
- [5] P. Biyani, K. Tsioutsoulouklis, and J. Blackmer, ““8 amazing secrets for getting more clicks”: Detecting clickbaits in news streams using article informality,” in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

- [6] Y. Chen, N. J. Conroy, and V. L. Rubin, “Misleading online content: Recognizing clickbait as false news,” in *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*, pp. 15–19, ACM, 2015.
- [7] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, “Fake news detection on social media: A data mining perspective,” *ACM SIGKDD explorations newsletter*, vol. 19, no. 1, pp. 22–36, 2017.
- [8] V. L. Rubin, N. Conroy, Y. Chen, and S. Cornwell, “Fake news or truth? using satirical cues to detect potentially misleading news,” in *Proceedings of the second workshop on computational approaches to deception detection*, pp. 7–17, 2016.
- [9] J. A. Caetano, J. F. de Oliveira, H. S. Lima, H. T. Marques-Neto, G. Magno, W. Meira Jr, and V. A. Almeida, “Analyzing and characterizing political discussions in whatsapp public groups,” *arXiv preprint arXiv:1804.00397*, 2018.
- [10] K. R. Canini, B. Suh, and P. L. Pirolli, “Finding credible information sources in social networks based on content and social structure,” in *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pp. 1–8, IEEE, 2011.
- [11] A. Gupta, H. Lamba, and P. Kumaraguru, “\$ 1.00 per rt# boston-marathon# prayforboston: Analyzing fake content on twitter,” in *2013 APWG eCrime researchers summit*, pp. 1–12, IEEE, 2013.

- [12] A. H. Wang, “Don’t follow me: Spam detection in twitter,” in *2010 international conference on security and cryptography (SECRYPT)*, pp. 1–10, IEEE, 2010.
- [13] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, “Detecting spammers on twitter,” in *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, vol. 6, p. 12, 2010.
- [14] S. Kaur, P. Kumar, and P. Kumaraguru, “Detecting clickbaits using two-phase hybrid cnn-lstm biterm model,” *Expert Systems with Applications*, p. 113350, 2020.
- [15] K. Garimella and G. Tyson, “Whatsapp doc? a first look at whatsapp public group data,” in *Twelfth International AAAI Conference on Web and Social Media*, 2018.
- [16] O. Batchelor, “Getting out the truth: the role of libraries in the fight against fake news,” *Reference Services Review*, vol. 45, no. 2, pp. 143–148, 2017.
- [17] T. Mitra and E. Gilbert, “Credbank: A large-scale social media corpus with associated credibility annotations,” in *ICWSM*, pp. 258–267, 2015.
- [18] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, “Protecting world leaders against deep fakes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 38–45, 2019.

- [19] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi, “Deep learning for deepfakes creation and detection,” *arXiv preprint arXiv:1909.11573*, vol. 1, 2019.
- [20] A. Chiatti, X. Yang, M. Brinberg, M. J. Cho, A. Gagneja, N. Ram, B. Reeves, and C. L. Giles, “Text extraction from smartphone screenshots to archive in situ media behavior,” in *Proceedings of the Knowledge Capture Conference*, pp. 1–4, 2017.
- [21] N. Ruchansky, S. Seo, and Y. Liu, “Csi: A hybrid deep model for fake news detection,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 797–806, ACM, 2017.
- [22] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, “Automatic detection of fake news,” *arXiv preprint arXiv:1708.07104*, 2017.
- [23] “News trends database.” [https : //bit.ly/2zVRLxK](https://bit.ly/2zVRLxK). Last accessed: 18-10-2017.
- [24] “Kaggle database.” [https : //bit.ly/2Ex5VsX](https://bit.ly/2Ex5VsX). Last accessed: 24-10-2017.
- [25] “Reuters database.” [https : //bit.ly/2BmqBQE](https://bit.ly/2BmqBQE). Last accessed: 22-10-2017.
- [26] S. Tariq, S. Lee, H. Kim, Y. Shin, and S. S. Woo, “Detecting both machine and human created fake face images in the wild,” in *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security*, pp. 81–87, ACM, 2018.

- [27] A. Agrawal, "Clickbait detection using deep learning," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, pp. 268–272, IEEE, 2016.
- [28] S. Yang, K. Shu, S. Wang, R. Gu, F. Wu, and H. Liu, "Unsupervised fake news detection on social media: A generative approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5644–5651, 2019.
- [29] W. Y. Wang, "'liar, liar pants on fire': A new benchmark dataset for fake news detection," *arXiv preprint arXiv:1705.00648*, 2017.
- [30] "Fake news corpus." [https : //github.com/several27/FakeNewsCorpus](https://github.com/several27/FakeNewsCorpus). Last accessed: 08-07-2020.
- [31] J. Nørregaard, B. D. Horne, and S. Adalı, "Nela-gt-2018: A large multi-labelled news dataset for the study of misinformation in news articles," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 13, pp. 630–638, 2019.
- [32] A. Zubiaga, M. Liakata, R. Procter, G. Wong Sak Hoi, and P. Tolmie, "Analysing how people orient to and spread rumours in social media by looking at conversational threads," *PloS one*, vol. 11, no. 3, p. e0150989, 2016.
- [33] E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. de Alfaro, "Some like it hoax: Automated fake news detection in social networks," *arXiv preprint arXiv:1704.07506*, 2017.
- [34] A. Pathak and R. K. Srihari, "Breaking! presenting fake news corpus for automated fact checking," in *Proceedings of the 57th Annual Meeting*

of the Association for Computational Linguistics: Student Research Workshop, pp. 357–362, 2019.

- [35] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, “Fakenews-net: A data repository with news content, social context and dynamic information for studying fake news on social media,” *arXiv preprint arXiv:1809.01286*, vol. 8, 2018.
- [36] F. K. A. Salem, R. Al Feel, S. Elbassuoni, M. Jaber, and M. Farah, “Fa-kes: A fake news dataset around the syrian war,” in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 13, pp. 573–582, 2019.
- [37] G. C. Santia and J. R. Williams, “Buzzface: A news veracity dataset with facebook user commentary and egos,” in *Twelfth International AAAI Conference on Web and Social Media*, 2018.
- [38] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, “An evaluation of popular copy-move forgery detection approaches,” *IEEE Transactions on information forensics and security*, vol. 7, no. 6, pp. 1841–1854, 2012.
- [39] S. Heller, L. Rossetto, and H. Schuldt, “The ps-battles dataset-an image collection for image manipulation detection,” *arXiv preprint arXiv:1804.04866*, 2018.
- [40] C. Boididou, S. Papadopoulos, M. Zampoglou, L. Apostolidis, O. Papadopoulou, and Y. Kompatsiaris, “Detection and visualization of misleading content on twitter,” *International Journal of Multimedia Information Retrieval*, vol. 7, no. 1, pp. 71–86, 2018.

- [41] X. Yang, Y. Li, and S. Lyu, “Exposing deep fakes using inconsistent head poses,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8261–8265, IEEE, 2019.
- [42] “Fakeapp.” <https://www.malavida.com/en/soft/fakeapp/>. Last accessed: 08-07-2020.
- [43] P. Korshunov and S. Marcel, “Deepfakes: a new threat to face recognition? assessment and detection,” *arXiv preprint arXiv:1812.08685*, 2018.
- [44] “Faceswap-gan.” Available in: <https://github.com/shaoanlu/faceswap> – GAN. Last accessed: 08-07-2020.
- [45] C. Sanderson and B. C. Lovell, “Multi-region probabilistic histograms for robust and scalable identity inference,” in *International conference on biometrics*, pp. 199–208, Springer, 2009.
- [46] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “Faceforensics++: Learning to detect manipulated facial images,” *arXiv preprint arXiv:1901.08971*, 2019.
- [47] “Faceswap github.” Available in: <https://github.com/deepfakes/faceswap>. Last accessed: 8-07-2020.
- [48] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, “The deepfake detection challenge (dfdc) preview dataset,” *arXiv preprint arXiv:1910.08854*, 2019.

- [49] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df: A large-scale challenging dataset for deepfake forensics,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3207–3216, 2020.
- [50] “B.s detector.” [https : //addons.mozilla.org/en – US/firefox/addon/bsdetector/](https://addons.mozilla.org/en-US/firefox/addon/bsdetector/). Last accessed: 17-07-2020.
- [51] “Surfsafe tool.” Available in: [https : //www.getsurfsafe.com/](https://www.getsurfsafe.com/). Last accessed: 07-07-2020.
- [52] “Fake news detector ai.” [http : //www.fakenewsai.com/](http://www.fakenewsai.com/). Last accessed: 07-07-2020.
- [53] “Trusted news.” [https : //trusted – news.com/](https://trusted-news.com/). Last accessed: 17-07-2020.
- [54] “Fake news detection.” [https : //fakenewsdetector.org/en](https://fakenewsdetector.org/en). Last accessed: 21-07-2020.
- [55] “Fake news guard.” Available in: [https : //www.fakenewsguard.com](https://www.fakenewsguard.com). Last accessed: 19-07-2020.
- [56] S. Volkova, K. Shaffer, J. Y. Jang, and N. Hodas, “Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on twitter,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, pp. 647–653, 2017.
- [57] “Fib by devpost.” [http : //ysnp.com/wp – content/wp – xmlrpcusr.php?y = fib – extension – chrome](http://ysnp.com/wp-content/wp-xmlrpcusr.php?y=fib-extension-chrome). Last accessed: 21-07-2020.

- [58] A. Bondielli and F. Marcelloni, “A survey on fake news and rumour detection techniques,” *Information Sciences*, vol. 497, pp. 38–55, 2019.
- [59] X. Zhou and R. Zafarani, “Fake news: A survey of research, detection methods, and opportunities,” *arXiv preprint arXiv:1812.00315*, vol. 2, 2018.
- [60] B. Markines, C. Cattuto, and F. Menczer, “Social spam detection,” in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 41–48, ACM, 2009.
- [61] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and M. Gonçalves, “Detecting spammers and content promoters in on-line video social networks,” in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 620–627, ACM, 2009.
- [62] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao, “Detecting and characterizing social spam campaigns,” in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pp. 35–47, ACM, 2010.
- [63] A. Gupta and P. Kumaraguru, “Credibility ranking of tweets during high impact events,” in *Proceedings of the 1st workshop on privacy and security in online social media*, p. 2, ACM, 2012.
- [64] F. Ahmed and M. Abulaish, “An mcl-based approach for spam profile detection in online social networks,” in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pp. 602–608, IEEE, 2012.

- [65] N. J. Conroy, V. L. Rubin, and Y. Chen, “Automatic deception detection: Methods for finding fake news,” *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015.
- [66] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, “Rumor has it: Identifying misinformation in microblogs,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1589–1599, Association for Computational Linguistics, 2011.
- [67] V. L. Rubin, N. J. Conroy, and Y. Chen, “Towards news verification: Deception detection methods for news discourse,” in *Hawaii International Conference on System Sciences*, 2015.
- [68] A. Aggarwal, S. Kumar, K. Bhargava, and P. Kumaraguru, “The follower count fallacy: Detecting twitter users with manipulated follower count,” 2018.
- [69] A. Magdy and N. Wanas, “Web-based statistical fact checking of textual documents,” in *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pp. 103–110, ACM, 2010.
- [70] S. Chhabra, A. Aggarwal, F. Benevenuto, and P. Kumaraguru, “Phi.sh/\$ocial: the phishing landscape through short urls,” in *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, pp. 92–101, ACM, 2011.
- [71] A. Aggarwal, A. Rajadesingan, and P. Kumaraguru, “Phishari: Automatic realtime phishing detection on twitter,” in *eCrime Researchers Summit (eCrime)*, 2012, pp. 1–12, IEEE, 2012.

- [72] A. Gupta, H. Lamba, and P. Kumaraguru, “\$1.00 per rt #boston-marathon #prayforboston: Analyzing fake content on twitter,” in *eCrime Researchers Summit (eCRS), 2013*, pp. 1–12, IEEE, 2013.
- [73] A. Gupta, H. Lamba, P. Kumaraguru, and A. Joshi, “Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy,” in *Proceedings of the 22nd international conference on World Wide Web*, pp. 729–736, ACM, 2013.
- [74] P. Dewan and P. Kumaraguru, “Towards automatic real time identification of malicious posts on facebook,” in *Privacy, Security and Trust (PST), 2015 13th Annual Conference on*, pp. 85–92, IEEE, 2015.
- [75] P. Dewan and P. Kumaraguru, “Facebook inspector (fbi): Towards automatic real-time detection of malicious content on facebook,” *Social Network Analysis and Mining*, vol. 7, no. 1, p. 15, 2017.
- [76] I. Sen, A. Aggarwal, S. Mian, S. Singh, P. Kumaraguru, and A. Datta, “Worth its weight in likes: Towards detecting fake likes on instagram,” in *Proceedings of the 10th ACM Conference on Web Science*, pp. 205–209, 2018.
- [77] I. Sen, A. Aggarwal, S. Mian, S. Singh, P. Kumaraguru, and A. Datta, “Worth its weight in likes: Towards detecting fake likes on instagram,” in *Proceedings of the 10th ACM Conference on Web Science*, pp. 205–209, ACM, 2018.
- [78] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, “Detecting rumors from microblogs with recurrent neural networks,” in *IJCAI*, pp. 3818–3824, 2016.

- [79] T. Pontes, M. Vasconcelos, J. Almeida, P. Kumaraguru, and V. Almeida, “We know where you live: privacy characterization of foursquare behavior,” in *Proceedings of the 2012 ACM conference on ubiquitous computing*, pp. 898–905, ACM, 2012.
- [80] T. Pontes, G. Magno, M. Vasconcelos, A. Gupta, J. Almeida, P. Kumaraguru, and V. Almeida, “Beware of what you share: Inferring home location in social networks,” in *Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on*, pp. 571–578, IEEE, 2012.
- [81] B. Vijgen *et al.*, “The listicle: An exploring research on an interesting shareable new media phenomenon,” *Studia Universitatis Babes-Bolyai-Ephemerides*, vol. 59, no. 1, pp. 103–122, 2014.
- [82] J. N. Blom and K. R. Hansen, “Click bait: Forward-reference as lure in online news headlines,” *Journal of Pragmatics*, vol. 76, pp. 87–100, 2015.
- [83] N. Ferro, F. Crestani, M.-F. Moens, J. Mothe, F. Silvestri, G. M. Di Nunzio, C. Hauff, and G. Silvello, *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, vol. 9626. Springer, 2016.
- [84] E. Gur and Z. Zelavsky, “Retrieval of rashi semi-cursive handwriting via fuzzy logic,” in *2012 International Conference on Frontiers in Handwriting Recognition*, pp. 354–359, IEEE, 2012.
- [85] S. Dutta, N. Sankaran, K. P. Sankar, and C. Jawahar, “Robust recognition of degraded documents using character n-grams,” in *2012 10th IAPR International Workshop on Document Analysis Systems*, pp. 130–134, IEEE, 2012.

- [86] A. AlSalman, A. El-Zaart, S. Al-Salman, and A. Gumaiei, "A novel approach for braille images segmentation," in *2012 International Conference on Multimedia Computing and Systems*, pp. 190–195, IEEE, 2012.
- [87] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition (alpr): A state-of-the-art review," *IEEE Transactions on circuits and systems for video technology*, vol. 23, no. 2, pp. 311–325, 2012.
- [88] J. Yang, K. Wang, J. Li, J. Jiao, and J. Xu, "A fast adaptive binarization method for complex scene images," in *2012 19th IEEE International Conference on Image Processing*, pp. 1889–1892, IEEE, 2012.
- [89] Y. Bassil and M. Alwani, "Ocr post-processing error correction algorithm using google online spelling suggestion," *arXiv preprint arXiv:1204.0191*, 2012.
- [90] K. Ntirogiannis, B. Gatos, and I. Pratikakis, "Performance evaluation methodology for historical document image binarization," *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 595–609, 2012.
- [91] S. Kowshik, V. Gautam, and K. Suganthi, "Assistance for visually impaired using finger-tip text reader using machine learning," in *2019 11th International Conference on Advanced Computing (ICoAC)*, pp. 7–12, IEEE, 2019.
- [92] V. Rajan and S. Raj, "Text detection and character extraction in natural scene images using fractional poisson model," in *2017 International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 1136–1141, IEEE, 2017.

- [93] R. Kushol, I. Ahsan, and M. N. Raihan, “An android-based useful text extraction framework using image and natural language processing,” *International Journal of Computer Theory and Engineering*, vol. 10, no. 3, 2018.
- [94] W. H. Gomaa, A. A. Fahmy, *et al.*, “A survey of text similarity approaches,” *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13–18, 2013.
- [95] N. Pradhan, M. Gyanchandani, and R. Wadhvani, “A review on text similarity technique used in ir and its application,” *International Journal of Computer Applications*, vol. 120, no. 9, 2015.
- [96] C. Samarinas and G. Tsoumakas, “Wamby: An information retrieval approach to web-based question answering,” in *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, pp. 1–8, 2018.
- [97] S. Kowser, S. Rahman, D. A. Shojol, and M. Kabir, *Study on Text Similarity Measure Algorithms For English Language*. PhD thesis, United International University, 2019.
- [98] C.-C. Hsu, Y.-X. Zhuang, and C.-Y. Lee, “Deep fake image detection based on pairwise learning,” *Applied Sciences*, vol. 10, no. 1, p. 370, 2020.
- [99] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “Learning to detect manipulated facial images,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1–11, 2019.

- [100] C.-C. Hsu, C.-Y. Lee, and Y.-X. Zhuang, “Learning to detect fake face images in the wild,” in *2018 International Symposium on Computer, Consumer and Control (IS3C)*, pp. 388–391, IEEE, 2018.
- [101] M. F. Hashmi, V. Anand, and A. G. Keskar, “A copy-move image forgery detection based on speeded up robust feature transform and wavelet transforms,” in *2014 international conference on computer and communication technology (ICCT)*, pp. 147–152, IEEE, 2014.
- [102] Z. Zhang, Z. Yu, and B. Su, “Detection of composite forged image,” in *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 11, pp. V11–572, IEEE, 2010.
- [103] H.-C. Hsu and M.-S. Wang, “Detection of copy-move forgery image using gabor descriptor,” in *Anti-counterfeiting, Security, and Identification*, pp. 1–4, IEEE, 2012.
- [104] S.-J. Ryu, M.-J. Lee, and H.-K. Lee, “Detection of copy-rotate-move forgery using zernike moments,” in *International workshop on information hiding*, pp. 51–65, Springer, 2010.
- [105] P. Kinghorn, L. Zhang, and L. Shao, “A region-based image caption generator with refined descriptions,” *Neurocomputing*, vol. 272, pp. 416–424, 2018.
- [106] Y. H. Tan and C. S. Chan, “Phrase-based image caption generator with hierarchical lstm network,” *Neurocomputing*, vol. 333, pp. 86–100, 2019.
- [107] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

- [108] B. Bayar and M. C. Stamm, “A deep learning approach to universal image manipulation detection using a new convolutional layer,” in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pp. 5–10, ACM, 2016.
- [109] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, “Two-stream neural networks for tampered face detection,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1831–1839, IEEE, 2017.
- [110] I. Korshunova, W. Shi, J. Dambre, and L. Theis, “Fast face-swap using convolutional neural networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3677–3685, 2017.
- [111] Y. Li, M. Chang, and S. Lyu, “In ictu oculi: Exposing ai created fake videos by detecting eye blinking,” in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, 2018.
- [112] D. Güera and E. J. Delp, “Deepfake video detection using recurrent neural networks,” in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, IEEE, 2018.
- [113] H. H. Nguyen, J. Yamagishi, and I. Echizen, “Capsule-forensics: Using capsule networks to detect forged images and videos,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2307–2311, IEEE, 2019.
- [114] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df: A new dataset for deepfake forensics,” *arXiv preprint arXiv:1909.12962*, 2019.

- [115] X. Yang, Y. Li, and S. Lyu, “Exposing deep fakes using inconsistent head poses,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8261–8265, IEEE, 2019.
- [116] A. Agarwal, R. Singh, M. Vatsa, and A. Noore, “Swapped! digital face presentation attack detection via weighted local magnitude pattern,” in *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pp. 659–665, IEEE, 2017.
- [117] Y. Zhang, L. Zheng, and V. L. Thing, “Automated face swapping and its detection,” in *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, pp. 15–19, IEEE, 2017.
- [118] Z. Akhtar and D. Dasgupta, “A comparative evaluation of local feature descriptors for deepfakes detection,” in *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pp. 1–5, IEEE, 2019.
- [119] R. Durall, M. Keuper, F.-J. Pfrendt, and J. Keuper, “Unmasking deepfakes with simple features,” *arXiv preprint arXiv:1911.00686*, 2019.
- [120] H. Mo, B. Chen, and W. Luo, “Fake faces identification via convolutional neural network,” in *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, pp. 43–47, 2018.
- [121] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua, “Towards open-set identity preserving face synthesis,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6713–6722, 2018.

- [122] N.-T. Do, I.-S. Na, and S.-H. Kim, “Forensics face detection from gans using convolutional neural network,” 2018.
- [123] F. Marra, D. Gragnaniello, D. Cozzolino, and L. Verdoliva, “Detection of gan-generated fake images over social networks,” in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pp. 384–389, IEEE, 2018.
- [124] D. Cozzolino, J. Thies, A. Rössler, C. Riess, M. Nießner, and L. Verdoliva, “Forensictransfer: Weakly-supervised domain adaptation for forgery detection,” *arXiv preprint arXiv:1812.02510*, 2018.
- [125] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, “Mesonet: a compact facial video forgery detection network,” in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, IEEE, 2018.
- [126] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, “Recurrent-convolution approach to deepfake detection-state-of-art results on faceforensics++,” *arXiv preprint arXiv:1905.00582*, 2019.
- [127] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, “Multi-task learning for detecting and segmenting manipulated facial images and videos,” *arXiv preprint arXiv:1906.06876*, 2019.
- [128] X. Ding, Z. Raziei, E. C. Larson, E. V. Olinick, P. Krueger, and M. Hahsler, “Swapped face detection using deep learning and subjective assessment,” *EURASIP Journal on Information Security*, vol. 2020, no. 1, pp. 1–12, 2020.

- [129] M. Du, S. Pentyala, Y. Li, and X. Hu, “Towards generalizable forgery detection with locality-aware autoencoder,” *arXiv preprint arXiv:1909.05999*, 2019.
- [130] J. Stehouwer, H. Dang, F. Liu, X. Liu, and A. Jain, “On the detection of digital face manipulation,” *arXiv preprint arXiv:1910.01717*, 2019.
- [131] T. Fernando, C. Fookes, S. Denman, and S. Sridharan, “Exploiting human social cognition for the detection of fake and fraudulent faces via memory networks,” *arXiv preprint arXiv:1911.07844*, 2019.
- [132] J. Li, T. Shen, W. Zhang, H. Ren, D. Zeng, and T. Mei, “Zooming into face forensics: A pixel-level analysis,” *arXiv preprint arXiv:1912.05790*, 2019.
- [133] X. Xuan, B. Peng, W. Wang, and J. Dong, “On the generalization of gan image forensics,” in *Chinese Conference on Biometric Recognition*, pp. 134–141, Springer, 2019.
- [134] X. Tu, H. Zhang, M. Xie, Y. Luo, Y. Zhang, and Z. Ma, “Deep transfer across domains for face antispooing,” *Journal of Electronic Imaging*, vol. 28, no. 4, p. 043001, 2019.
- [135] U. A. Ciftci and I. Demir, “Fakecatcher: Detection of synthetic portrait videos using biological signals,” *arXiv preprint arXiv:1901.02212*, 2019.
- [136] Y. Li and S. Lyu, “Exposing deepfake videos by detecting face warping artifacts,” *arXiv preprint arXiv:1811.00656*, 2018.
- [137] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo, “Deepfake video detection through optical flow based cnn,” in *Proceedings of the IEEE*

International Conference on Computer Vision Workshops, pp. 0–0, 2019.

- [138] P. Korshunov and S. Marcel, “Speaker inconsistency detection in tampered video,” in *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2375–2379, IEEE, 2018.
- [139] P. Korshunov, M. Halstead, D. Castan, M. Graciarena, M. McLaren, B. Burns, A. Lawson, and S. Marcel, “Tampered speaker inconsistency detection with phonetically aware audio-visual features,” in *International Conference on Machine Learning*, no. CONF, 2019.
- [140] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros, “Everybody dance now,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5933–5942, 2019.
- [141] N. Yu, L. S. Davis, and M. Fritz, “Attributing fake images to gans: Learning and analyzing gan fingerprints,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7556–7566, 2019.
- [142] R. Wang, L. Ma, F. Juefei-Xu, X. Xie, J. Wang, and Y. Liu, “Fakespotter: A simple baseline for spotting ai-synthesized fake faces,” *arXiv preprint arXiv:1909.06122*, 2019.
- [143] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo, “Face x-ray for more general face forgery detection,” *arXiv preprint arXiv:1912.13458*, 2019.
- [144] “Fake news on whatsapp.” [http : //bit.ly/2miuv9j](http://bit.ly/2miuv9j). Last accessed: 27-08-2019.

- [145] P. Jain and P. Kumaraguru, “On the dynamics of username changing behavior on twitter,” in *Proceedings of the 3rd IKDD Conference on Data Science, 2016*, p. 6, ACM, 2016.
- [146] S. M. SIRAJUDEEN, N. F. A. AZMI, and A. I. ABUBAKAR, “On-line fake news detection algorithm,” *Journal of Theoretical & Applied Information Technology*, vol. 95, no. 17, 2017.
- [147] A. Gupta and P. Kumaraguru, “Twitter explodes with activity in mumbai blasts! a lifeline or an unmonitored daemon in the lurking?,” tech. rep., 2012.
- [148] A. Alahmadi, A. Joorabchi, and A. E. Mahdi, “A new text representation scheme combining bag-of-words and bag-of-concepts approaches for automatic text classification,” in *2013 7th IEEE GCC Conference and Exhibition (GCC)*, pp. 108–113, IEEE, 2013.
- [149] S. Z. Mishu and S. Rafiuddin, “Performance analysis of supervised machine learning algorithms for text classification,” in *Computer and Information Technology (ICCIT), 2016 19th International Conference on*, pp. 409–413, IEEE, 2016.
- [150] E. E. G. Tan and B. Ang, “Clickbait: Fake news and role of the state,” 2017.
- [151] B. Gardiner, “You’ll be outraged at how easy it was to get you to click on this headline,” *Wired Magazine*, 2015.
- [152] B. D. Horne and S. Adali, “This just in: fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire

- than real news,” in *Eleventh International AAAI Conference on Web and Social Media*, 2017.
- [153] A. Elyashar, J. Bendahan, and R. Puzis, “Detecting clickbait in on-line social media: You won’t believe how we did it,” *arXiv preprint arXiv:1710.06699*, 2017.
 - [154] M. Potthast, S. Köpsel, B. Stein, and M. Hagen, “Clickbait detection,” in *European Conference on Information Retrieval*, pp. 810–817, Springer, 2016.
 - [155] X. Cao, T. Le, *et al.*, “Machine learning based detection of clickbait posts in social media,” *arXiv preprint arXiv:1710.01977*, 2017.
 - [156] M. M. U. Rony, N. Hassan, and M. Yousuf, “Diving deep into clickbaits: Who use them to what extents in which topics with what effects?,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ASONAM ’17, (New York, NY, USA), pp. 232–239, ACM, 2017.
 - [157] S. R. Khater, O. H. Al-sahlee, D. M. Daoud, and M. El-Seoud, “Clickbait detection,” in *Proceedings of the 7th International Conference on Software and Information Engineering*, pp. 111–115, ACM, 2018.
 - [158] R. Pochampally and V. Varma, “User context as a source of topic retrieval in twitter,” in *Workshop on Enriching Information Retrieval (with ACM SIGIR)*, pp. 1–3, Citeseer, 2011.
 - [159] “Octoparse.” *https : //www.octoparse.com/download*. Last accessed: 09-02-2020.

- [160] “Subreddits world news.” *https : //www.reddit.com/r/worldnews*. Last accessed: 21-04-2019.
- [161] “Subreddits news.” *https : //www.reddit.com/r/news*. Last accessed: 22-04-2019.
- [162] X. Wang, W. Jiang, and Z. Luo, “Combination of convolutional and recurrent neural network for sentiment analysis of short texts,” in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 2428–2437, 2016.
- [163] X. Yan, J. Guo, Y. Lan, and X. Cheng, “A biterm topic model for short texts,” in *Proceedings of the 22nd international conference on World Wide Web*, pp. 1445–1456, ACM, 2013.
- [164] S. Kaur, P. Kumar, and P. Kumaraguru, “Automating fake news detection system using multi-level voting model,” *Soft Computing*, 2019.
- [165] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *Journal of machine learning research*, vol. 12, no. Aug, pp. 2493–2537, 2011.
- [166] D. Mulfari, A. Celesti, M. Fazio, M. Villari, and A. Puliafito, “Using google cloud vision in assistive technology scenarios,” in *2016 IEEE Symposium on Computers and Communication (ISCC)*, pp. 214–219, IEEE, 2016.
- [167] A. Sboev, T. Litvinova, D. Gudovskikh, R. Rybka, and I. Moloshnikov, “Machine learning models of text categorization by author gender using topic-independent features,” *Procedia Computer Science*, vol. 101, pp. 135–142, 2016.

- [168] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, pp. 3111–3119, 2013.
- [169] A. Kulkarni and A. Shivananda, “Exploring and processing text data,” in *Natural Language Processing Recipes*, pp. 37–65, Springer, 2019.
- [170] D. S. Sisodia, “Ensemble learning approach for clickbait detection using article headline features,” *Informing Science: The International Journal of an Emerging Transdiscipline*, vol. 22, pp. 031–044, 2019.
- [171] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.
- [172] Y. Kim, “Convolutional neural networks for sentence classification,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1746–1751, 2014.
- [173] L. Eidnes, “Auto-generating clickbait with recurrent neural networks,” 2015.
- [174] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

- [175] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” *arXiv preprint arXiv:1611.03530*, 2016.
- [176] M. Hoffman, F. R. Bach, and D. M. Blei, “Online learning for latent dirichlet allocation,” in *advances in neural information processing systems*, pp. 856–864, 2010.
- [177] T. M. Kodinariya and P. R. Makwana, “Review on determining number of cluster in k-means clustering,” *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.
- [178] J. R. Finkel, T. Grenager, and C. Manning, “Incorporating non-local information into information extraction systems by gibbs sampling,” in *Proceedings of the 43rd annual meeting on association for computational linguistics*, pp. 363–370, Association for Computational Linguistics, 2005.
- [179] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, “Theano: A cpu and gpu math compiler in python,” in *Proc. 9th Python in Science Conf*, vol. 1, pp. 3–10, 2010.
- [180] P. Vithlani and C. Kumbharana, “Comparative study of character recognition tools,” *International Journal of Computer Applications*, vol. 118, no. 9, 2015.
- [181] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum, “Optimizing semantic coherence in topic models,” in *Proceedings of the conference on empirical methods in natural language processing*, pp. 262–272, Association for Computational Linguistics, 2011.

- [182] D. López-Sánchez, J. R. Herrero, A. G. Arrieta, and J. M. Corchado, “Hybridizing metric learning and case-based reasoning for adaptable clickbait detection,” *Applied Intelligence*, pp. 1–16, 2017.
- [183] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467*, 2016.
- [184] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, “Detection of cyberbullying incidents on the instagram social network,” *arXiv preprint arXiv:1503.03909*, 2015.
- [185] C. Vondrick, H. Pirsiavash, and A. Torralba, “Generating videos with scene dynamics,” in *Advances In Neural Information Processing Systems*, pp. 613–621, 2016.
- [186] D. tool. *https : //github.com/deepfakes/faceswap*. Last accessed: 21-10-2019.
- [187] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “Faceforensics: A large-scale video dataset for forgery detection in human faces,” *arXiv preprint arXiv:1803.09179*, 2018.
- [188] G. Goswami, M. Vatsa, and R. Singh, “Rgb-d face recognition with texture and attribute features,” *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 10, pp. 1629–1640, 2014.
- [189] N. Abudarham and G. Yovel, “Reverse engineering the face space: Discovering the critical features for face identification,” *Journal of vision*, vol. 16, no. 3, pp. 40–40, 2016.

- [190] L. K. Hansen and P. Salamon, “Neural network ensembles,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 993–1001, 1990.
- [191] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.
- [192] DeepfaceLab. [http : //bit.ly/2Px44sU](http://bit.ly/2Px44sU). Last accessed: 28-09-2019.
- [193] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.
- [194] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, “Action recognition in video sequences using deep bi-directional lstm with cnn features,” *IEEE Access*, vol. 6, pp. 1155–1166, 2017.
- [195] Z. Xu, S. Li, and W. Deng, “Learning temporal features using lstm-cnn architecture for face anti-spoofing,” in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 141–145, IEEE, 2015.
- [196] M. Sivaram, V. Porkodi, A. S. Mohammed, and V. Manikandan, “Detection of accurate facial detection using hybrid deep convolutional recurrent neural network.,” *ICTACT Journal on Soft Computing*, vol. 9, no. 2, 2019.
- [197] H. H. Nguyen, J. Yamagishi, and I. Echizen, “Use of a capsule network to detect fake images and videos,” *arXiv preprint arXiv:1910.12467*, 2019.

- [198] P. Korshunov and S. Marcel, “Vulnerability assessment and detection of deepfake videos,” in *The 12th IAPR International Conference on Biometrics (ICB)*, pp. 1–6, 2019.
- [199] “Google cloud vision api.” <https://cloud.google.com/vision/docs/libraries>. Last accessed: 07-05-2020.
- [200] A. Mishra, K. Alahari, and C. V. Jawahar, “Scene text recognition using higher order language priors,” in *BMVC*, 2012.
- [201] A. Biørn-Hansen, T. A. Majchrzak, and T.-M. Grønli, “Progressive web apps: The possible web-native unifier for mobile development,” in *International Conference on Web Information Systems and Technologies*, vol. 2, pp. 344–351, SCITEPRESS, 2017.