

Framework for Efficient Spam Detection in Online Social Network

A thesis

submitted in the fulfillment of the requirement for the award of the
degree of

Doctor of Philosophy

Submitted by

Sanjeev Rao

(Registration No. : 951703012)

Under the supervision of

Dr. Anil Kumar Verma

(Professor, CSED)

Dr. Tarunpreet Bhatia

(Assistant Professor, CSED)



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA-147004, PUNJAB, INDIA

June 2023

Certificate

I hereby declare that the work being presented in this thesis, "Framework for Efficient Spam Detection in Online Social Network", in fulfillment of the requirements for the award of degree of DOCTOR OF PHILOSOPHY submitted in Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my work carried out under the supervision of Dr. Anil Kumar Verma, Professor and Dr. Tarunpreet Bhatia, Assistant Professor, Computer Science and Engineering Department and refers other researcher works which are duly listed in the reference section. The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.



(Sanjeev Rao)

Registration No. 951703012

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



(Dr. Anil Kumar Verma)

Professor

Computer Science and Engineering Department

Supervisor



(Dr. Tarunpreet Bhatia)

Assistant Professor

Computer Science and Engineering Department

Supervisor

Acknowledgements

I owe to the grace of the almighty, whose divine light provides me with the perseverance, guidance, patience, inspiration, faith, and strength to carry out this work.

I want to express my earnest gratitude to my supervisor, Dr. Anil Kumar Verma, Professor and Dr. Tarunpreet Bhatia, Assistant Professor, Computer Science and Engineering Department, for their invaluable guidance, motivation, constructive comments and support. Their enthusiasm has motivated me to reach my goal and improved the quality of my Ph.D. work. I am incredibly indebted to them for providing time to listen and give their valuable suggestions. I want to thank Dr. Ponnurangam Kumaraguru (PK), IIIT, Hyderabad, for his guidance and support.

I am grateful to Dr. Shalini Batra, Professor and Head, Computer Science and Engineering Department. I would like to give my sincere thanks to Dr. Maninder Singh, Professor and Dean of Academic Affairs (DoAA), for his valuable support during my Ph.D. tenure. This thesis was not possible without his support and cooperation.

I am thankful to the Doctoral committee comprising Dr. Maninder Kaur, Assistant Professor, Dr. Nidhi Kalra, Assistant Professor, Computer Science and Engineering Department, and Dr. Kulbir Singh, Professor, Electronics and Communication Engineering Department, for monitoring the progress and providing valuable suggestions for the improvement of this work. I thank all the Computer Science and Engineering Department faculty members and staff members for their co-operation and support.

Also, I want to thank my dearest friends Nitin, Sahil, Vijay, Jasminder, Parampreet, and Jasmeet for their inspiration, support, and cooperation. The discussions with them were always valuable and motivating during this journey.

Words are inadequate in paying regards to my father Late. Shri Jai Bhagwan Rao, mother Mrs. Ram Murti Rao, and my brother Rajiv Rao for their love, blessings, continuous encouragement, and patience. I express deep gratitude to my wife Surbhi Rishi, beloved Shrishti and Shivam, whose love, care, patience, and support have been a tremendous encouragement to me.

I would like to thank all who contributed directly or indirectly during this journey of research work to make my Ph.D. a successful event.



(Sanjeev Rao)

Abstract

Online Social Networks (OSNs) are perpetually evolving and used in plenteous applications such as communication, news, entertainment, businesses, gaming, marketing and advertisement, live-streaming, job search, dating, education, healthcare, etc. Simultaneously, cybercriminals and botnets with groups of fake/bot accounts use OSNs to disseminate spam, misleading facts, fake news, hate speech, and malicious links to targeted users or masses to perform cyber-crimes, earn money, polarize sentiments, and impact users' online interaction time. Moreover, prevalent spam degrades available information quality, network bandwidth, computing power, and speed. Recently, AI-enabled Deepfakes have exacerbated these issues at large. Thus, to detect and eliminate social spam and spammers from OSNs, it is necessary to review recent research on these topics.

This doctoral thesis thoroughly reviews existing solutions for social spam and spammer detection techniques. Initially, background related to social spam, the spamming process, and social spam taxonomy is discussed. Later, the extensive review reveals various essentials and critical challenges to detect and combat social spam. The thesis uncovers important information about features used, dimensionality reduction techniques used for feature selection/extraction, existing datasets, and various machine learning and deep learning methodologies used for social spam and spammer detection, along with their strengths and limitations. Also, the thesis explores information related to recent AI-enabled Deepfake (text, image, and video) spam and its countermeasures.

The doctoral thesis aims to advance the field of spam detection in OSN by developing Machine Learning (ML) and Deep Learning (DL) based approaches to address the most pressing issues in social spam detection, thereby improving the performance of spam detection. Most previous research relied on small datasets and witnessed class imbalance issues, resulting in biased outcomes towards the majority class. This study uses data-sampling techniques such as NearMiss and SmoteTomek to address the class imbalance problem. Traditional word representation techniques are inefficient and time-consuming when generating contextual word vectors. The relevant features are

extracted in this study using recent word-representation/embedding techniques before feeding any ML/DL model.

In this study, a framework for social spam detection is proposed using ML and DL-based approaches to improve the performance of social spam detection. For ML-based approaches, two voting ensemble models are proposed. Initially, nine baseline ML models are trained and tested on imbalanced and balanced datasets. Later, the models are ensemble using hard and soft voting mechanisms. After the model evaluation on test data, the best parameter values are extracted, and the best prediction model is finalized by adjusting the parameters. In the Proposed Hard Voting Ensemble (PHVE), the final class prediction is made using the majority voting of each classifier. However, in the Proposed Soft Voting Ensemble (PSVE), the target label with the highest sum of weighted probability is chosen as the final prediction. Finally, it is revealed that the performance of the PHVE and PSVE models outperforms other ML baseline models over the balanced datasets.

For DL-based approaches, two hybrid approaches are proposed using advanced pre-trained word embedding techniques, deep learning approaches, and self-attention mechanism on a balanced *combined dataset* retrieved using the SmoteTomek data sampling technique. The embeddings are generated using recent GloVe and FastText pre-trained word embeddings and passed into the deep neural network comprised of Conv1D and Bi-directional recurrent neural network layers with the self-attention mechanism for improved contextual understanding and effective results. Experiments and comparisons show that the proposed hybrid framework with deep learning-based approaches outperformed other techniques/approaches.

Furthermore, meticulous discussions, existing challenges, and emerging issues such as robustness of detection systems, scalability, real-time datasets, evade strategies used by spammers, coordinated inauthentic behavior, and adversarial attacks on machine learning-based spam detectors, etc., have been discussed with possible directions for future research.

Keywords: Data sampling techniques, Deep Learning, Machine Learning, Natural Language Processing, Online Social Network, Social Spam

Abbreviations

ACO	Ant Colony Optimization
ADAM	Adaptive Moment Optimizer
ADASYN	Adaptive Synthetic Sampling
ADB	AdaBoost Classifier
AE	Auto Encoders
AI	Artificial Intelligence
AML	Adversarial Machine Learning
ANN	Artificial Neural Network
API	Application Programming Interface
APTs	Advanced Persistent Threats
AUC	Area Under the Curve
BC	Bagging Classifier
BERT	Bi-directional Encoder Representations from Transformers
Bi-GRU	Bi-directional Gated Recurrent Unit
Bi-LSTM	Bi-directional Long Short-Term Memory Network
BN	Bayesian Network
BoW	Bag of Words
BP-MLL	Back-Propagation Multi-Label Learning

CAN	Convolutional Attention Network
CFS	Correlation-based Feature Selection
ChiSqr	Chi-Squared
CIB	Coordinated Inauthentic Behavior
CNN	Convolutional Neural Network
COMM	Community Detection
CON	Consistency Subset Evaluation
DBN	Deep Belief Network
DDoS	Distributed Denial of Service
DFDC	Deepfake Detection Challenge
DL	Deep Learning
DNN	Deep Neural Network
DQL	Deep Q-Learning
DT	Decision Tree
DWT	Discrete Wavelet Transform
EDA	Exploratory Data Analysis
EL	Ensemble Learning
ELM	Extreme Learning Machine
ELMo	Embeddings from Language Models
ETC	ExtraTree Classifier
FCB-SA	FastText+Conv1D+Bi-GRU+Bi-LSTM with Self-Attention Mechanism

FDNN	Feed-Forward Deep Neural Network
FND	Fake News Dataset
FNR	False Negative Ratio
FPR	False Positive Rate
GAN	Generative Adversarial Networks
GBC	Gradient Boosting Classifier
GCB-SA	GloVe+Conv1D+Bi-GRU+Bi-LSTM with Self-Attention Mechanism
GloVe	Global Vectors
GNN	Generative Convolutional Neural Networks
GPT	Generative Pre-trained Transformer
GPU	Graphical Processing Unit
GRU	Gated Recurrent Unit
HPMS	Hierarchical Meta-Path Score
IDS	Intrusion Detection System
IDSS	Intelligent Decision Support System
INB	Incremental Naïve Bayes
Infogain	Information Gain
IPFS	Inter Planetary File System
IQM	Image Quality Metrics
KNN	K Nearest Neighbor
LCRN	Long-term Recurrent Convolution Network

LIWC	Linguistic Inquiry and Word Count
LR	Logistic Regression
LSH	Locality Sensitive Hashing
LSTM	Long Short-Term Memory Network
LSVM	Linear Support Vector Machine
LTRSB	Learning to Rank Social Bot framework
MAE	Mean Absolute Error
MCC	Mathew Correlation Coefficient
ML	Machine Learning
MLP	Multi-Layer Perceptron
MLKNN	Multi-Label k-Nearest Neighbors
MNB	Multinomial Naïve Bayes
MoA	Metaheuristic Optimization Algorithms
MOEFS	Multi-Objective Evolutionary Feature Selection
MSE	Mean Square Error
MVMC	Multi-View Modularity Clustering
MVSD	Multi-View Learning for Social Spammer Detection
NB	Naïve Bayes
NLP	Natural Language Processing
NM	NearMiss
NMF	Negative Matrix Factorization

OOV	Out of Vocabulary
OSN	Online Social Network
OWASP	Open Web Applications Protection Project
PCA	Principle Component Analysis
PDS	Probabilistic Data Structures
PHVE	Proposed Hard Voting Ensemble
POS	Parts of Speech
PPV	Positive Predictive Value
PSO	Particle Swarm Optimization
PSVE	Proposed Soft Voting Ensemble
RAKEL	Random K-label Subsets
RDNN	Regularized Deep Neural Network (RDNN)
RF	Random Forest
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
ROC	Receiver Operator Characteristic
ROS	Random Over Sampling
RST	Rough Set Theory algorithm
SA	Self-Attention
SCNN	Semantic Convolutional Neural Network
SGD	Stochastic Gradient Descent

SL	Supervised Learning
SMOTE	Synthetic Minority Over-sampling Technique
SMFSR	Supervised Matrix Factorization Technique using Social Regularization
SNA	Social Network Analysis
SNS	Social Network Sites
SSCL	Sequential Stacked CNN and LSTM model
SSL	Semi-Supervised Learning
S³D	Semi-Supervised Spam Detection
ST	SmoteTomek
SVM	Support Vector Machine
TF-IDF	Term Frequency/Inverse Document Frequency
TNR	True Negative Rate
TPR	True Positive Rate
TRAC	Trolling, Aggression, and Cyberbullying
TSP	Traid Significance Profile
TSVM	Transductive Support Vector Machine
URL	Uniform Resource Locator
USL	Unsupervised Learning
WOA	Whale Optimization Algorithm
XGB	Extreme Gradient Boosting
XSS	Cross-Site Scripting

Notations and Symbols

σ	Sigma Activation Function
A	Accuracy
AV	Accuracy Variation
b_c	Training Bias Matrix
b_f	Bias Matrix of Forget Gate
c_t	Current Memory Cell State at time step t
c_{t-1}	Previous Memory Cell State at time step t
$\tilde{c}_t$	New Candidate Value Vector
$\sqrt{d_k}$	Scaling Factor
f_t	Forget Gate at time step t
F	F1 Score/Measure
FN	False Negative
FP	False Positive
FPR	False Positive Rate
$h_{t (backward)}$	Backward LSTM
$h_{t (forward)}$	Forward LSTM
h_t	Current state at time step t
h_{t-1}	Previous state at time step t

i_t	Input Gate at time step t
o_t	Output Gate at time step t
K	Key Vector Matrix
KS	Klout Score
P	Precision
Q	Query Vector Matrix
$\tanh$	hyperbolic tangent Activation Function
TN	True Negative
TP	True Positive
TPR	True Positive Rate
R	Recall
r_t	Reset Gate at time step t
V	Value Vector Matrix
W_C	Training Weight Matrix
W_f	Weight Matrix of Forget Gate
W_i	Weight Matrix of Input Gate
W_r	Weight Matrix of Reset Gate
W_z	Weight Matrix of Update Gate
X_t	Current input at time step t
z_t	Update Gate at time step t

List of Publications

International Journals

1. Rao, S., Verma, A. K., & Bhatia, T. (2021). A review on social spam detection: challenges, open issues, and future directions. *Expert Systems with Applications*, Elsevier, 186, 115742. [I.F. 8.665]. SCIE. (Published)
2. Rao, S., Verma, A. K., & Bhatia, T. (2023). Hybrid Ensemble Framework with Self-Attention Mechanism for Social Spam Detection on Imbalanced Data. *Expert Systems with Applications*, Elsevier, 217, 119594. [I.F. 8.665]. SCIE. (Published)

Book Chapters

3. Rao, S., Verma, A.K. and Bhatia, T., 2021. Online Social Networks Misuse, Cyber Crimes, and Counter Mechanisms. In *Analyzing Global Social Media Consumption* (pp. 183-203). IGI Global. SCOPUS. (Published)
4. Rao, S., Verma, A.K. and Bhatia, T., 2021. Evolving Cyber Threats, Combating Techniques, and Open Issues in Online Social Networks. In *Handbook of Research on Cyber Crime and Information Privacy* (pp. 219-235). IGI Global. SCOPUS. (Published)

Conference

5. Rao, S., Verma, A.K. and Bhatia, T., 2018. Online Social Network: Challenges, Combating Techniques and Future Research Directions”, In *International Conference on Computational Intelligence and Data Analytics (ICCIDA-2018)* at GIFT, Bhubaneswar. Odisha. (Presented)

Table of Contents

Certificate.....	i
Acknowledgements.....	ii
Abstract	iii
Abbreviations.....	v
Notations and Symbols	xi
List of Publications	xiii
Table of Contents.....	xiv
List of Figures	xvii
List of Tables	xix
Chapter 1 : Introduction	1
1.1 Online Social Network and its Misuse	1
1.2 Spam and Spamming Process in OSNs: An Overview	3
1.2.1 Social Spam	3
1.2.2 Evolution of Spam	5
1.2.3 Social Spamming Process	6
1.3 Challenges and Open Issues to Detect Spam in OSN.....	13
1.4 Research Motivation.....	19
1.5 Research Objectives.....	20
1.6 Thesis Contribution.....	20
1.7 Thesis Organization	22
Chapter 2 : Literature Review	25
2.1 Traditional Social Spam Detection Techniques.....	26
2.1.1 URL List-Based Spam Filtering Techniques	26
2.1.2 Honeypot and HoneyNet-Based Spam Detection Techniques.....	27

2.2 Machine Learning and Deep Learning Approaches used for Spam Detection.....	28
2.2.1 Text Spam Detection Approaches.....	34
2.2.2 Word Representation Techniques used for Text Spam Detection	45
2.2.3 Pre-trained Word Embedding Techniques	46
2.2.4 AI-Enabled Multimedia Spam (Deepfakes) Detection Approaches	50
2.3 Spammer Account Detection Approaches	60
2.3.1 Fake/Compromised and Cloned/Sybil Account Detection	60
2.3.2 Spambot Account Detection	67
2.3.3 Spammer Community Detection.....	74
2.4 Summary.....	77
Chapter 3 : Proposed PHVE and PSVE Models using Machine Learning.....	79
3.1 Introduction.....	79
3.2 Related work	80
3.3 Contributions	82
3.4 Proposed Methodology	83
3.4.1 Dataset Acquisition.....	84
3.4.2 Dataset Balancing	85
3.4.3 Dataset Pre-processing.....	87
3.4.4 Exploratory Data Analysis	88
3.4.5 Baseline Machine Learning Models	91
3.4.6 Proposed PHVE and PSVE Models.....	92
3.5 Experimental Setup and Results	95
3.5.1 Dataset Description.....	95
3.5.2 Experiment Setting.....	96
3.5.4 Results and Discussions	96
3.5.5 Comparison with Existing Studies.....	104
3.6 Summary.....	105
Chapter 4 : Proposed GCB-SA and FCB-SA approaches using Deep Learning.....	106
4.1 Introduction.....	106

4.2 Related Work	107
4.3 Contribution	108
4.4 Proposed Methodology	109
4.4.1 Baseline DL Models	109
4.4.2 Proposed GCB-SA and FCB-SA Approaches	114
4.5 Experiment Setup and Results	118
4.5.1 Dataset description	118
4.5.2 Experimental Settings	118
4.5.3 Results and Discussions	120
4.5.4 Comparison with existing studies	123
4.6 Summary	124
Chapter 5 : Conclusions and Future Scope	126
5.1 Conclusions	126
5.2 Future Scope	128
References	129

List of Figures

Figure 1.1: Relationship between web 2.0, social media and OSNs	1
Figure 1.2: Evolution of spam from e-mail spam to Deepfakes.	5
Figure 1.3: Spamming process in OSNs	6
Figure 1.4: Taxonomy of social spam.....	7
Figure 1.5: AML scenario exemplifying poison, evasion, and exploratory attack.....	14
Figure 2.1: Taxonomy of various social spam detection techniques and approaches .	25
Figure 2.2: General ML/DL-based methodology used for social spam detection.....	29
Figure 2.3: Confusion matrix.....	31
Figure 2.4: Word2Vec word embedding with a CBOW and skip-gram model.....	47
Figure 2.5: Deepfake creation process using autoencoders and decoders	52
Figure 3.1: Proposed framework with ML and DL approaches for social spam detection.....	83
Figure 3.2: Word Cloud for <i>ham</i> words.....	89
Figure 3.3: Word Cloud for <i>spam</i> words	89
Figure 3.4: Frequency of <i>ham</i> and <i>spam</i> messages over the length of the message....	89
Figure 3.5: Top 20 most common words used in text messages	90
Figure 3.6: Schematic for proposed hard voting ensemble (PHVE) and soft voting ensemble (PSVE) models	93
Figure 3.7: Consolidated accuracy analysis of proposed ensemble models (PHVE and PSVE) with other good-performing ML models on different imbalanced and balanced datasets.	103
Figure 4.1: Simple architecture of RNN	110
Figure 4.2: Architecture of LSTM	110
Figure 4.3: Architecture of GRU	112

Figure 4.4: Schematic for proposed GCB-SA and FCB-SA approaches.....	115
Figure 4.5: Comparison of training accuracy and validation accuracy for proposed hybrid DL-based approaches	122
Figure 4.6: Comparison of training loss and validation loss for proposed hybrid DL- based approaches	122
Figure 4.7: Consolidated accuracy analysis of proposed hybrid DL-based approaches with DL baseline approaches.....	123

List of Tables

Table 1.1: Definitions of social spam and OSN platform manipulation.....	3
Table 1.2: Facts about social spam and fake accounts.....	4
Table 2.1: Dimensionality reduction techniques used for social spam detection	31
Table 2.2: Summary of various feature analyses used for social spam detection.....	32
Table 2.3: Performance measures used for social spam detection.....	33
Table 2.4: Summary of text spam detection approaches used in OSNs	41
Table 2.5: Comparison of word representation and word embedding techniques.....	49
Table 2.6: Summary of AI-enabled multimedia spam detection approaches	56
Table 2.7: Summary of Fake and Sybil/cloned account detection approaches.....	64
Table 2.8: Summary of spambot account detection approaches.....	71
Table 2.9: Summary of spammer community detection approaches	75
Table 3.1: Sample of combined dataset	85
Table 3.2: Summary of individual datasets and combined dataset along with CIR	86
Table 3.3: Shape of imbalanced datasets before and after using data balancing techniques	87
Table 3.4: Statistics for the combined dataset	90
Table 3.5: Parameter values for ML models.....	96
Table 3.6: Performance comparison of nine ML baselines with proposed voting ensemble models on imbalanced and balanced SMS spam dataset.....	98
Table 3.7: Performance comparison of nine ML baselines with proposed voting ensemble models on imbalanced and balanced Ling spam dataset	99
Table 3.8: Performance comparison of nine ML baselines with proposed voting ensemble	

models on balanced Fake news dataset	100
Table 3.9: Performance comparison of various ML baselines with proposed voting ensemble models on the imbalanced combined dataset, balanced combined dataset retrieved using NM and ST data sampling technique.....	102
Table 3.10: Performance comparison of PHVE and PSVE with existing ML-based approaches	104
Table 4.1: Summary of the proposed GCB-SA approach.....	117
Table 4.2: Summary of the proposed FCB-SA approach	118
Table 4.3: Optimised hyperparameter settings used for building and testing proposed hybrid DL-based approaches	119
Table 4.4: Performance comparison of various DL baseline approaches with proposed hybrid DL-based approaches	120
Table 4.5: Performance comparison of proposed hybrid DL-based approaches with existing approaches	124

Chapter 1 : Introduction

1.1 Online Social Network and its Misuse

The outburst of the World Wide Web (WWW) is the most productive and expedient way to find and distribute information. Web 2.0 has emerged from "read-only web" to "read-write web" constitutes tools such as podcasting, blogging, tagging, RSS feed, social bookmarking, social networking platforms, web content voting, etc. With the arrival of Web 2.0, the media has become very dynamic and used as blogs, messenger services, websites, social networking sites, etc., to provide vast applications such as communication, news, entertainment, businesses, gaming, marketing and advertisement, live-streaming, job search, dating, education, healthcare, etc. Among many social media tools, the popularity and usage of Online Social Networks (OSNs) have increased incredibly and become essential for every user worldwide. The relationship between web 2.0, social media, and online social networks is shown in Figure 1.1.

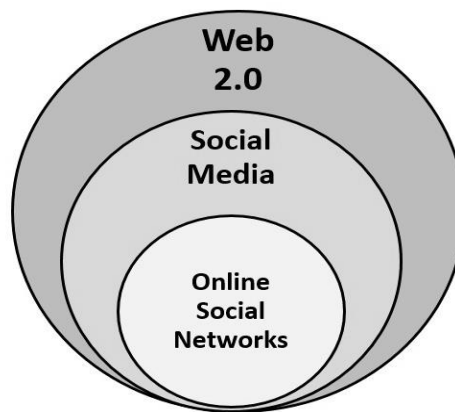


Figure 1.1: Relationship between web 2.0, social media and OSNs

Some work/contents of this chapter have been peer-reviewed and published in:

- Rao, S., Verma, A. K., & Bhatia, T. (2021). A review on social spam detection: challenges, open issues, and future directions. Expert Systems with Applications, Elsevier, 186, 115742. [I.F. 8.665]. SCIE.

OSNs are nowadays an indispensable tool for communication on account of its rise, simplicity, and efficacy. OSNs are essential for establishing social or professional relationships and interactions, expanding corporate reputation, promotions, understanding public opinions, news propagation, online learning, disaster management, gaming, healthcare, recommender systems, etc. OSN platform enables users to engage online by sending and accepting friend invitations, posting and sharing content, comments, likes, dislikes, tags, posts, followers, following, etc. [1]. Facebook, YouTube, Twitter, Sina-Weibo, Myspace, Tumblr, Google+, Snapchat, Instagram, etc., have billions of regular users. Facebook is the most popular OSN, with 2.89 billion active users. [2].

At the same time, many innocent users sporadically expose their personal information to strangers on OSNs. Cyber-criminals such as spammers, hackers, and other third-party corporations/apps utilize such personal information to perform various kinds of cyber-crimes like social engineering attacks, spamming, extortion-malware, drug trafficking, misinformation, cyberbullying, hijacking clicks, identity theft, phishing, mistrust, fake profiles, and cyber espionage, etc. Through convenient access to technologies, false information can be disseminated rapidly to millions of users on OSNs, which can often take the form of big campaigns and social movement warfare against any nation and administration. Medical misinformation and unverifiable content concerning the COVID-19 pandemic are circulating at an alarming rate on social media. It is vital to minimize the advent of rumors and misleading facts during the COVID-19 crisis as it causes people panic, apprehension, and distress, leading them to develop psychiatric illnesses [3]. Several forms of misinformation, such as spam, fake news, rumors, trolls, fake stories, political conspiracies, Deepfakes, etc., have become an offensive threat to public discourse, businesses, and nations. Deepfakes produced by Artificial Intelligence (AI) and Deep Learning (DL) have recently become widespread. AI and DL can generate and alter videos with the faces and voices of innocent actors, legislators, superstars, and other well-known personalities [4], [5].

Rao et al. [6], [7] discussed information related to various cyber-crimes associated with OSNs and OSN vulnerabilities. The Open Web Applications Protection Project (OWASP) study found top-ten Web vulnerabilities [8]. MITRE ATT&CK offered a publicly

accessible knowledge base as an attack matrix to counter cyber-attacks, threats, vulnerabilities, etc. [9].

In OSNs, the developers and designers have focused much on the design and features. However, less emphasis is provided on the privacy and security of OSN users. So this has become a wide area of research and has fascinated the interest of many research scholars.

1.2 Spam and Spamming Process in OSNs: An Overview

1.2.1 Social Spam

The term “Spam” refers to unwelcome messages in the form of bulk messages, text, images, malicious tiny URLs, trending hashtags, video clips, notifications, friends/follow requests, porn videos, phone numbers, healthcare advice, ransomware, stock market spams, forged commercials, fake reviews/comments, fake news, and political astroturfing, etc. by spammers to make good earnings. The spam content prevalent on specifically online social network sites or websites with user-generated content is known as Social spam. In addition, the definitions of social spam and OSN platform manipulation according to Twitter and Facebook are mentioned in Table 1.1.

Table 1.1: Definitions of social spam and OSN platform manipulation

OSN Platform	Social spam and OSN platform manipulation description
Twitter [10]	- Spamming is a barred action that disrupts Twitter policies, involving several methods of platform manipulation and efforts such as bulk, violent, or deceptive activities to fool or betray users. Furthermore, platform manipulation can be accompanied by the following:
	- Commercially-driven spam to push Twitter traffic/conversations to other accounts, websites, products, services, initiatives, etc.
	- Using trending hashtags and malicious clickjacking URLs makes inauthentic activities to style accounts or content more prevalent.
	- Coordinated activities through fake accounts or bots to artificially influence conversations.

Facebook [11]	- Spam involves targeting individuals with inappropriate content or requests. These include sending bulk messages, sharing unnecessary links or photos on Facebook user accounts, and sending friend requests for friends to people you don't know personally. - Spam is often distributed by clicking on malicious links or downloading malware. Spammers often obtain access to the Facebook accounts of people who are then used to send out spam.
------------------	--

Some important facts about annoying social spam and spammer accounts collected from various cybersecurity reports are given in Table 1.2.

Table 1.2: Facts about social spam and fake accounts

Cybersecurity reports	Facts
Nexgate [12]	In 2013, social media spam escalated by 355%. Facebook has 100 times more spam and four times more phishing attacks than any other OSN. YouTube contains five times more risky content than any other OSN. Fake profiles spread more spam as compared to legitimate accounts. Five out of seven new OSN profiles are spam profiles. 15% of social spam includes URLs leading to spam content, pornography, malware, etc. For every 200 messages on social media, one comprises spam.
Statista	Social media users noticed an approximately 47% rise in spam in their OSN environment [13]. Approximately 16% of all Facebook user profiles are false or cloned [14].
Spamhaus	The U.S., China, Vietnam, India, and Germany were among the top five spam-originated nations in 2017 [15].
Trend Micro	YouTube and Twitter enforced a ban on fake and manipulated Deepfakes [16]
New York Times	Published an article regarding proliferating misinformation on social media regarding the coronavirus infodemic related to facemasks, sanitizers, other healthcare advice, etc. [17].
Facebook	In Q1 2020, 1.9 billion spam posts were taken down from Facebook. Facebook removed 507 fake accounts, 131 Instagram accounts, 146 pages, and 78 network groups worldwide, indulging in inauthentic behavior [18]. Facebook has removed 837M spam content, 583M fake accounts, 21M nudity or porn content, 3.5M graphic violence, and 2.5M hate speech in the first quarter of 2018 [19].

1.2.2 Evolution of Spam

Spreading unwanted messages in the form of text, images, URLs, hashtags, news, video clips, notifications, friends/follow requests, etc., in bulk using electronic messaging systems (e-mail, Instant messaging systems, SMS, etc.) is one of the earliest spam practices which emerged from 1978 onwards on the Internet [20]. In 1990, instant messaging spam (SPIM) was developed on instant messaging (IM) platforms such as Skype, Yahoo messenger, AOL, etc. Simultaneously, web search engines witness spam known as Spamdexing, i.e., careful manipulation of search engine indexes. Spamdexing involves content spam, tag spam, link farming, cloaking links, click frauds, and pharming attacks to raise the rank of the target website using search engine optimization (SEO) techniques and page ranking algorithms [21], [22]. The evolution of spam (1978–2020 onward) from e-mail spam to AI-enabled multimedia spam is given in Figure 1.2.

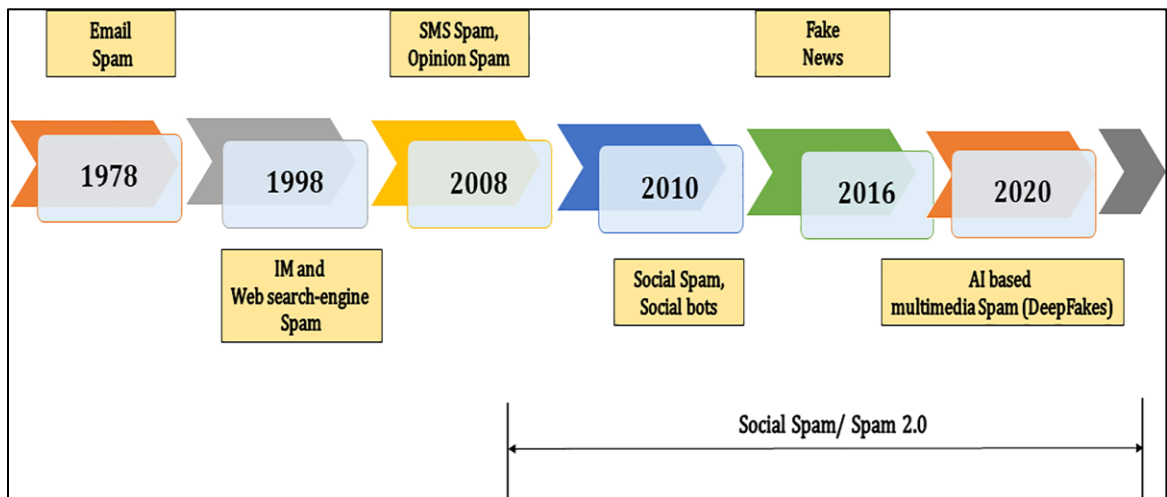


Figure 1.2: Evolution of spam from e-mail spam to Deepfakes.

The arrival of interactive Web 2.0 in the initial 2000s, where users could generate and share their content, brought a new form of spam, i.e., Spam 2.0 or social spam refers to undesirable, malevolent, unsolicited content or behavior, offered in diverse ways, comprising Wikipedia spam, mobile/SMS spam, microblogs spam (SPLOG), tweets, malicious links, fake news, fake reviews/comments (opinion spam), etc. These include spam prevalent on mobile phones, social media, blogs, OSNs, video sharing sites, online gaming platforms, social e-commerce, professional networking websites, etc. [23], [24]. In the last decade, spam has evolved into new forms like social spambots, misinformation

through fake accounts/websites, and AI-enabled multimedia spam. Ferrara [25] articulates the growth of established digital spam on the Internet.

1.2.3 Social Spamming Process

The widespread spam content is available on OSNs as text, malicious links, images, Deepfakes, etc. Fake accounts, spambots, crowdsourcing platforms, and cloned accounts are significant contributors for spreading such spam. In addition, these spamming accounts initiate spam propagation and involve online users with spam to do fraudulent activities. These accounts use link farming, click-baiting, clickjacking, cloaking, redirection, etc., to develop spam in OSN environments. The spamming process in OSNs works in three steps, as presented in Figure 1.3.

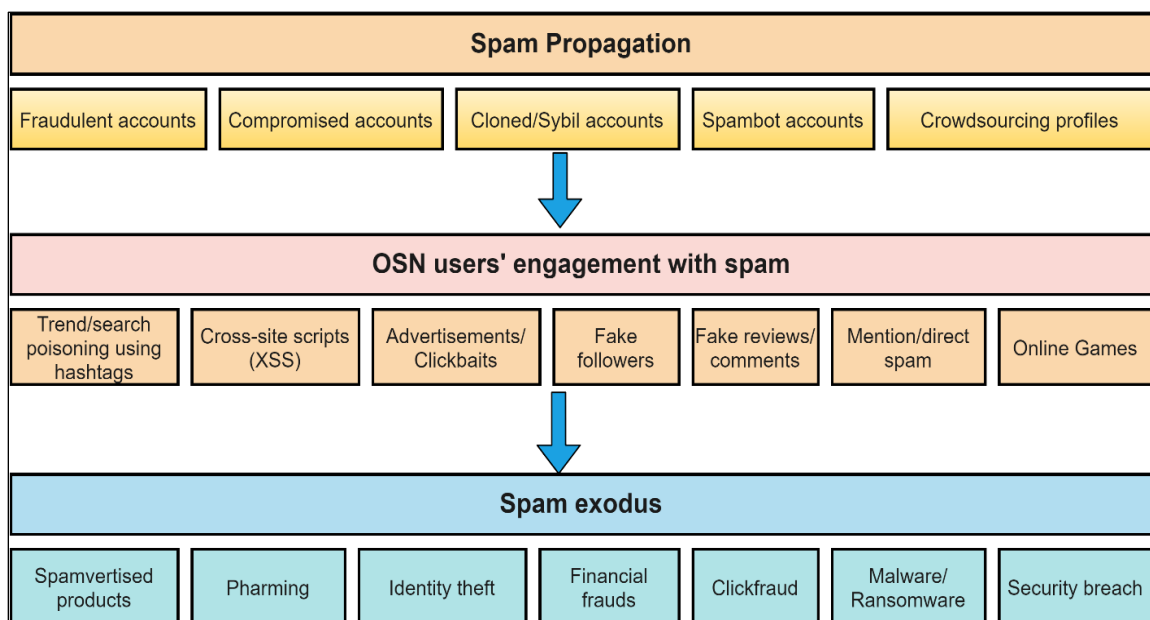


Figure 1.3: Spamming process in OSNs

a) Spam propagation: Extensive spamming is not an individual effort; it can be accomplished through a group of spammers such as fake accounts, compromised accounts, cloned/Sybil accounts, social spambots, Crowdturfing profiles, etc., with various spam propagation tactics [25]. The social spam taxonomy comprising various social spam and spammer accounts is depicted in Figure 1.4. The brief information about different spammer accounts is discussed below:

- **Fake profiles:** Often, malevolent users create numerous counterfeit accounts for spamming and phishing accomplishments and influence a more significant number of trustworthy users and groups [26]. Cybercriminals use bots to capture the target user's private data by sending them friend requests. Once the target user approves the friend request, all its personal information is visible to a fake user, otherwise hidden from the external user. Also, fake profiles have been used to post fake reviews in bulk to escalate product trade revenues or service offerings [27].

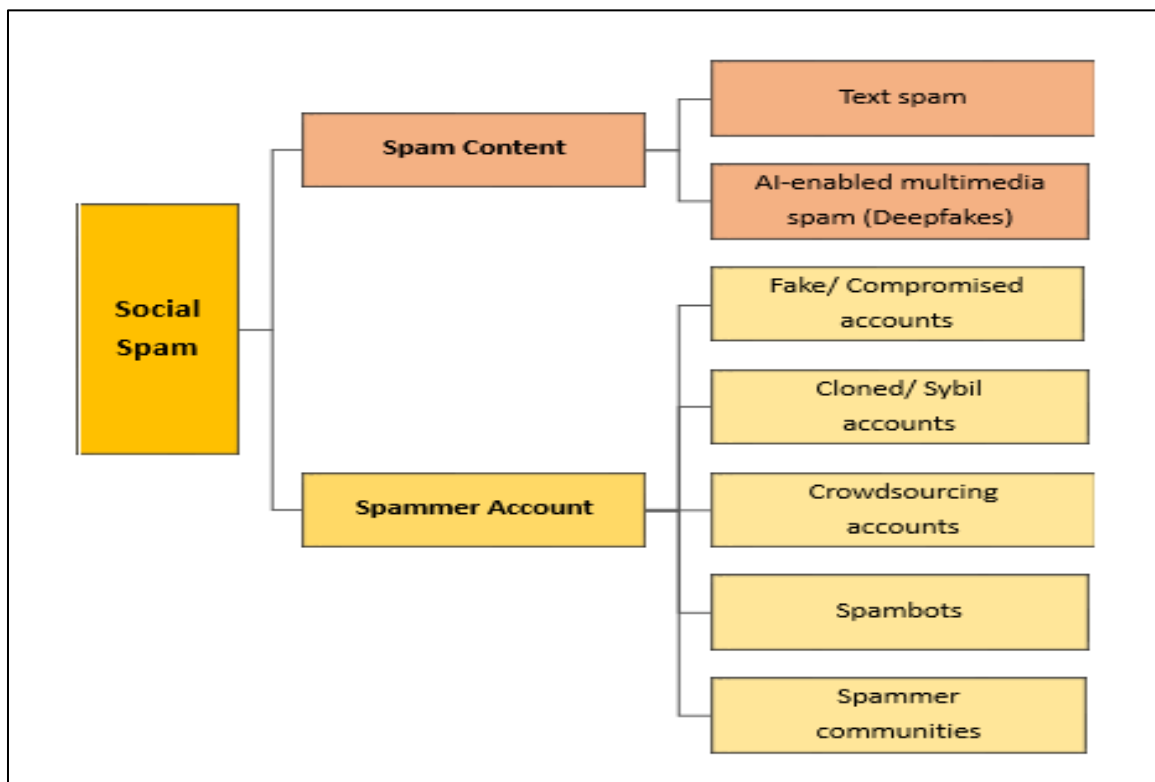


Figure 1.4: Taxonomy of social spam

- **Compromised accounts:** These are the genuine accounts that cyber-criminals have controlled to perform criminal activities such as sending fake friend requests, spreading misinformation and malicious content, etc., on their social network. Compromised accounts are hard to detect as they mimic legitimate users in OSNs [28], [29].
- **Cloned/Sybil accounts:** Sometimes, a legitimate user with a non-malicious intention creates duplicate profiles to manage different social circles. These profiles could be exploited to influence genuine user profiles, sending advertisements, profile browsing,

messaging, multiple friend/follower requests, likes, comments, etc. [30]. Sybil accounts have been used to create pseudonymous identities [31], i.e., multiple fake identities used to pollute the OSN network with malicious and fake information. Sybil's community can be a list of accounts made by the same user or similar intent by many users [32], [33]. These accounts have been used to influence networks through surplus voting and repeated messaging in OSNs to propagate political agendas, election campaigning, manipulating likes, views, vote count, etc. [34].

- **Social bots/Socbots/Spambots:** Social bots, also known as socbots, are automated programs/agents in the social botnet to interact on social media platforms to influence the course of discussions and readers' views by propelling spam to an enormous number of users in a concurrent manner. A social botnet is a collection of social bots handled by a solo botmaster (Command and Control- C&C software) who works together to perform malicious activities such as spamming while imitating legitimate OSN users to decrease their risk of detection. The botmaster can direct all bots directly to post spam that can be perceived by all its followers [35]. Social bots are compromised and controlled by human accounts, or cyber-criminals programmed social bots to undertake various adversary activities [36]–[38]. Social bots are perfect if used to disseminate something useful. Still, they are sometimes misused by cyber-criminals as essential tools for propagating polluted content in the form of trolls, fake news, spam and initiating Distributed Denial of Service (DDoS) attacks [39]–[41]. Social bots have been designed to build news headlines automatically, influence public opinion, boost popularity with increasing posts, likes, fake friends, and followers to spread business, health advertisements [29] and, deceive through pay-per-click [42], and propagate religious campaigns [43], terrorist propaganda and recruitments [44], etc. Chatbots are used in instant messaging systems to spread spam, fake likes/views/comments/news, etc.; influence bots influence users of multiple social media sites by stimulating discussions on a specific matter, political propaganda, etc. [45]. Social bots can affect public sentiment by spreading disinformation through political astroturfing. Social bots distributed thousands of tweets during the 2010 U.S. midterm elections to guide users to websites with fake stories that endorse particular candidates

[46]. Another example is an assault of 25,860 social bots distributing 440793 tweets to disrupt the Russian election conversations [47].

- **Crowdturfing:** It is an amalgam of two approaches, ‘crowdsourcing’ and ‘astroturfing.’ Crowdsourcing is a marketing strategy in which a company outsources some of its operational tasks to an ambiguous group, such as Freelancer [48] and Sandahl [49]. Amazon Mechanical Turk employs freelancers for commercials, spamming, and swaying people by raising the number of followers, friends, etc. [50]. In contrast, astroturfing is the deceptive procedure of delivering an orchestral advertisement or public relations campaign under the pretext of false approval from fake ratings/voting, or comments [51]. Also, crowdsourcing accounts are used to abuse or expand the social reputation of personalities. These spammers have a certain level of similarity with automated accounts or bots but are not that consistent [52].

b) OSN users’ engagement with spam: Spammers use various strategies to entice and engage OSN users to click on something they have not intended to click. User behavior in OSN is important in establishing network trust. OSN users can be classified as highly active and less-active users depending upon their engagement with OSN. For cybercriminals, highly engaged users are more influential and vulnerable to cyber-attacks and reverse-engineering attacks due to their level of attention and extensive network than less-active users. Several ways to sway online users are trend/search poisoning famous hashtags, malicious tiny URLs, Cross-Site Scripts (XSS), mouth-watering commercials (product images, lottery messages, healthcare product spam, porn images, and video clips, etc.), clickbait news headlines, hoaxes, artificial media (Deepfakes), online games (spam notifications, video ads, banners, micropayments links, etc.), like buttons, bogus reviews/comments, fake followers requests for fame [53], direct/mention spam (uninvited mention/replies to OSN user), follow spam and reposting, etc. [7].

c) Spam exodus/departure: After luring OSN users into spam engagement, spam finally exodus to perform social engineering attacks. Social engineering attacks are psychological manipulations in which users are enticed into making security mistakes or leaking sensitive

information [54], [55]. Some of the substantial social engineering attacks associated with social spam and human behavior in OSNs are mentioned below:

- **Spamvertising:** This entails mass advertising products/services/offers through unscrupulous advertising activities generally seen in the form of viagra, e-cigarettes, lottery tickets, pornographic links, online jobs claiming easy money, etc.
- **Phishing:** Phishing is the most prevalent form of social engineering attack. Phishing can be categorized as vishing (voice phishing), smishing (SMS phishing), spear phishing (target individuals), whaling (targets high-ranking victims within the organization), etc. The majority of phishing attacks aim to accomplish three goals, i.e., first, to utilize deceptive tiny links to route users to suspicious websites that host phishing landing pages, second to acquire personal information such as names, passwords, contact numbers, and credit card details, etc., third to utilize threats, panic, and an urgency situation to compel the user into acting promptly [7], [55].
- **Pharming:** Pharming is used for fraud traffic redirection. Cybercriminals steal victims' personal information by installing harmful malware on victims' computers to redirect them to fake websites without their permission, resulting in identity theft, data theft, and financial fraud. Pharming has become a significant threat to e-commerce and online banking organizations [6].
- **Clickbaiting and clickjacking:** Cybercriminals lure people into clicking on certain news headlines/topics to redirect them to a different webpage that has nothing to do with the news headlines to generate enormous income from misdirected links/advertisements [6], [54]. In clickjacking, cybercriminals trap users into clicking on some link, image, button, or text to capture their OSN session to perform further financial fraud [7], [56].
- **Cyberbullying:** A cyberbullying assault is meant to annoy OSN users, mainly adolescents and young females, through catfishing, sexting, stalking, trolling, pornographic links, revenge porn, blackmailing, violent content, etc. Online gaming platforms detect a rise of

39% in assaults in the year 2020 [57]. Often, both text and voice chat are bombarded with offensive comments. The long-term repercussions of cyberbullying include anxiety, panic, rage, and despair among such OSN users [58]–[60].

- **Ransomware attack:** This is an extortion method in which an attacker encrypts data on a victim's computer and then demands money to decode it. Scareware ransomware employed by pirated apps was used to extort money to restore the program or fix the infection. Ransomware attacks cost \$7.5 billion in 2019 [57]. The WannaCry ransomware has recently targeted the British healthcare system [61].
- **Distributed Denial of Service (DDoS) Attack:** A DDoS assault is a deceptive attempt to disrupt the regular traffic of a targeted website, device, or network by flooding the target or its surrounding systems with Internet traffic. The most common DDoS attacks are TCP synchronized flood attacks, ping of death attacks, buffer overload, smurf attacks, teardrop attacks, etc. [62]. Many prominent OSNs are vulnerable to bots, Sybil identities, fraudulent and compromised accounts, and other forms of organized attack [60], [63].
- **Cyber espionage/spy:** Cybercriminals employ malware, hacking, and proxy servers to steal private information such as studies, patents, customers' databanks, employees' details, trade secrets, etc., and to ruin company's and nation's financial reputations without their awareness. Secret agencies also exploit social network data to detect and predict crime, violence, riots, and other similar events [64], [65].
- **Coordinated inauthentic behavior (CIB):** The inauthentic behavior of online entities such as a group of OSN users, pages, apps, websites, bots, fake accounts, etc., coordinated to promote a given product/service or communication to mislead readers or followers intentionally in an organized way. A group of cybercriminals organizes collective actions to boost the social media attention metrics (likes, retweets, reposts, share count, preferences, etc.) on a given topic/product or to alter the course of a narrative [18], [66], [67].

- **Attribute inference attacks:** Attribute inference attacks in OSNs are done to find users' information that has not been made public, such as employment, friends list, religion, gender, etc. Attackers can use inferred attributes to link online users across various websites or create detailed and composite user profiles, increasing privacy and security risk. These attacks can be grouped into two categories. The first category of attribute inference can be accomplished through non-network features such as text and images from profiles and user posts. Cybercriminals, third-party apps, and advertisement companies apply ML/DL-based recommender systems to execute privacy attacks on users by stealing their data, such as profile information, pictures, likes/dislikes, preferences, location, behavior, friends/followers information, etc. Later, the stolen data and inferred characteristics can be leveraged to execute other assaults, such as spear phishing and cloning attacks, that may further escalate privacy and security issues [68], [69]. Another category of attribute inference is based on homophily to infer the latent attributes and links using network characteristics and social relations. The network characteristics-based attribute inference is made to discover the correlation between network characteristics and latent features using social network analysis (SNA) tools. The reach of nodes was considered to be an important factor in inferring user demographics and social roles. The attacker can infer other users' attributes and additional sensitive information from known users' attributes, friendship links, group membership attributes, and social network graphs [29].

- **Organization mining attack:** Cybercriminals obtain sensitive data of a company's personnel via OSNs to conduct an organization mining attack. Sometimes, malicious users can send multiple friend requests to company employees or attack a specific employee in the targeted organization by leveraging organizational social network data. Such assault can be applied to friend-based inference methodologies and within organizations to discover influential employees and their hidden links to form new teams [70]. Sánchez-Corcuera et al. [71] described how cyber-criminals could collect employees' confidential data via Twitter to initiate an attack on an organization by locating highly designated personnel within the organization. After gathering information about the company, cybercriminals create superfluous profiles to impersonate an employee. They also showed

how the centralities derived from nodes in a directed network affect the inference of user roles in organizations.

Due to the evolving existence of social spam, it is challenging to monitor and control spam on OSNs [72], [73]. However, researchers have introduced several counter-spam methods, such as URL list-based filtering methods, honeynet/pot-based methods, etc. ML and DL-based frameworks have recently been trending due to their advantages over traditional approaches. The Literature review chapter presents a detailed review of some important existing studies for social spam detection.

1.3 Challenges and Open Issues to Detect Spam in OSN

Some of the challenges and open issues related to spam detection in OSN are discussed below:

- **Real-time data collection, pre-processing, and data analysis:** The acquisition and analysis of a large, unbiased dataset covering content and user account details for real-time social spam detection are challenging. Social data enables the organization to gather and process data from various sources, such as user tweets, blog posts, status updates, etc. It allows companies to recognize and address customer responses as quickly as possible. Unlike Twitter Streaming API, some OSNs, such as Facebook Graph API, have no access to real-time streaming, which could otherwise allow the analytical approach to analyze OSN in real-time. Thereby analyzing social data in real-time remains an attractive challenge. Therefore, new efficient techniques and algorithms to investigate data streams with less time and memory consumption are still important.
- **Scalability:** Scalability is a significant problem in evaluating any ML/DL-based model's effectiveness because of the ever-growing OSN data. Further, the ever-increasing OSN size brought other challenges, such as managing massive users, apprising users' profiles, walls, status, network traffic, data storage administration, growing infrastructure administration, operational costs, etc. AI-based algorithms and Bigdata techniques are promising solutions to OSN scalability issues [74]–[76].

- **Multidimensional features:** To boost precision and protect other susceptible aspects, features evaluation from several dimensions such as temporal, writing styles, semantic, statistical, and so on in a single technique is required. Since users are expected to preserve uniformity in various functions, it is essential to incorporate a thriving selection of features and ranking that will operate with each user independently.
- **Class imbalance and spam drift:** Most ML approaches suggested have used a balanced dataset with positive and negative samples. The inherent design of ML algorithms works best for the main non-spam class, whereas ignoring the minority (spam) class. The unequal distribution of spam classes significantly reduces classification efficiency. Spam tweets change characteristics over time as spammers regularly change their attack tactics to escape spam detectors. This drastically decreases the performance of spam detectors dependent on ML, which must be addressed. Incorporating ensemble, semi-supervised, and meta-heuristic optimization algorithms with existing frameworks can solve class imbalance and spam drift problems [77], [78]. Yet, more research is required for class imbalance and spam drift challenges.
- **Adversarial ML attacks:** Adversaries can launch multifarious attacks during learning or evaluation (test). Adversaries can pollute the training data to provide the wrong classification by classifier (poisoning attack), generate adverse samples during testing to avoid detection (evasion attack), and collect sensitive training data via a learning model (privacy attack). Because of these adverse behaviors at the design level, spam detectors at OSNs are prone to various adversary attacks. Adversarial examples are produced by making small perturbations to the input to considerably increase the ML model's loss. Recent studies of adversaries' attacks reveal that text classifiers are easily deceived by adversaries that substantially alter text wording, structure, or spelling [79]. An eloquent AML example is shown in Figure 1.5 with several adversarial attacks, i.e., causative/poisoning, evasion, and exploratory/inference attack based on adversary behavior. In causative/poison attacks, the adversary gathers data to train and alter the learner's classifier. The learner subsequently uses the modified adversary data to train their classifier. As a result, the learner learns the wrong classification method for classification training, including false positives and negatives.

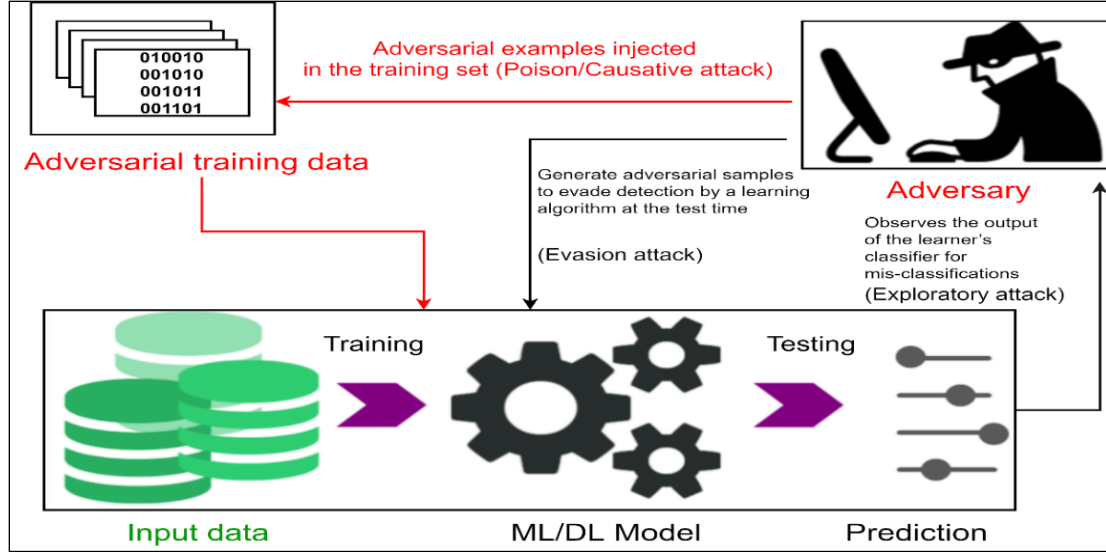


Figure 1.5: AML scenario exemplifying poison, evasion, and exploratory attack

In an evasion attack, adversaries have no direct access to the training data; instead, they generate adversarial samples via a learning algorithm to evade detection by causing misclassifications by the classifier model. In an exploratory/inference attack, the adversary perceives the classifier output from various data and attempts to identify vulnerabilities (e.g., what data the learners misclassify).

Further, it creates adverse data that utilizes these vulnerabilities to improve the classifier's misclassification rate during testing or evaluation. Therefore, it has been observed that a more complex model that studies the previous encounters between the learner and the adversary will benefit the learner by determining the adversary's behavior more accurately and changing its classifier accordingly. The adversaries performed several security violations, such as integrity, availability, and privacy, by manipulating OSN system functionalities. Fast heuristics-based algorithms to predict adverse behavior reliably within quantifiable limits will lead to more precise and robust models. Eradicating adulterated samples using model sanitization and multiple classifiers are considered defensive solutions in causative/poison attacks. The defense mechanisms for exploratory attacks involve hiding important functionalities from adversaries and fooling adversaries with randomized feedback mechanisms. Transfer learning can also be effective in adverse attacks after determining the similarity between source and target domain datasets [80]. Adversarial examples can be leveraged to improve ML models' performance or robustness

[81]. Few research papers have addressed the adversarial problem [79], [81]–[86], but it is still a critical challenge.

- **Transfer Learning:** Transfer learning is to reuse a model built for a task as the basis of a model for a second task. Previous knowledge of one domain and task is passed into another domain and task. Transfer learning can be achieved using either the develop-model approach or pre-trained models. In NLP, transfer learning techniques mainly focus on pre-trained language modeling that redesigns and reuses DL models in high-resource languages and domains. ML models are data-intensive and require significant data to build good NLP models. The requisite annotated data is unavailable beforehand for certain domains or languages. These annotations are time-consuming, costly exercises, and a challenge. Transfer learning through pre-trained language models is a viable approach in this context. Some available pre-trained models that can work with image and text-based data are Oxford models - VGG-16/19, Microsoft's ResNet50, DenseNet, Google's Inception, and Word2Vec model, and Stanford's GloVe Model, etc. [87]. These pre-trained models can be used with other recent DL models that expect such data to be trained [88].
- **Word embedding:** Word embedding is the initial processing layer in the DL model. In word embedding, words are denoted as low-dimensional vectors that grasp the text corpus's syntax and semantics. Similar-meaning words tend to occur in a similar context. Some of the word embedding techniques are embedding's from Language Models (ELMo), Word2Vec, GloVe, FastText, and Bidirectional Encoder Representations from Transformers (BERT) can be incorporated into new research advances [89], [90].
- **Advanced sentiment analysis:** An enormous effort has been made towards the study of human behavior for opinion/sentiment analysis, but still, some critical problems can't be tackled, such as multimodal content, domain dependency, sarcasm/slang detection, non-English, unreliable and inadequate data, fake reviews/comments, and subjective bias detection, etc. [90]–[95]. Future research should be carried out to improve the analysis to work with varied linguistic and non-English features from real-time OSN posts.
- **Robustness:** Several other shortcomings, such as automated and fast feature engineering, processing capabilities, network traffic, latency, stego images (images with embedded

text/URLs) and videos, etc., can be solved with advanced AI techniques, DL techniques, Extreme Learning Machine (ELM), NLP, image processing, etc. The advanced DL techniques can help to develop a faster and more efficient system that can work with real and unbiased data collected from various sources such as streaming OSN API, continuous attributes, language insensitive patterns, effective sampling strategies, personomies in the tag-based system, pornographic spammers, stego images and Deepfakes, etc.

- **Deepfakes:** The neural network models such as GPT-2/3 and image generation models such as BigGAN, StyleGAN, CycleGAN, etc., are used to create, manipulate, and style images and videos. The Deepfakes can be misused to spread disinformation [96], [97]. Hence, anti-Deepfake techniques and mechanisms need to be developed. Blockchain-based frameworks can track Deepfakes by allowing the system to check the fake video's validity. Some research papers [4], [96]–[101] recently published towards the neural network paradigm for the development and detection of synthetic media (Deepfakes). However, it is still a crucial problem to address.
- **Antispam plugins:** Antispam plugins can be a software mechanism or browser extension to block spam from accessing a device. The anti-spam software uses protocols to determine unsolicited and inappropriate messages and hold such messages from a user's inbox or wall. Besides, these plugins block spam, spam reporting, quarantine spam, multiple spamming account tracking, personal whitelist management, etc. Spam content and Deepfakes can be controlled by providing awareness and making legal rules, policies, and voluntary actions. Cybersecurity agencies and social network sites must provide secure and trusted OSN platforms free from digital pollution. Efficient anti-Deepfake and anti-spambot mechanisms can be developed to track, combat, and thwart such spam content [72], [102], [103].
- **Trust-based recommender system:** Privacy and trust are vital considerations when OSN detects spam, i.e., how trust between users and the OSN ecosystem influences the distribution of information. Trust can be evaluated on different aspects, such as analyzing users' forwarding behavior, similarities between two users, topic sensitivity and evidence theory methods, etc. Further, new factors may be combined with the reward and penalty

scheme in neural networks and reinforcement learning-based trust-based analysis. Anonymous information should be used best to protect user privacy [104]–[107]. The recommender system helps OSN users to approve fake friend requests, among many friend requests. Evaluating large texts on OSN and websites is challenging, although it has become a valued source of sentiments on content such as profiles, posts, comments, reviews, products, services, events, etc. [108], [109]. An efficient recommender system is required to recommend genuine sentiments about profiles, products, reviews, events, etc.

- **Information diffusion:** Social bots in OSNs are crucial in disseminating deterrent information such as fake news, political campaigns, biased behavior, commercials, healthcare infodemic, marketing, crimes, etc. As a result, multi-edge nodes facilitate decision-making by analyzing the graph of connections between users on social networks and analyzing influential users. Consequently, it is important to investigate how the OSN propagates information. Information diffusion implicates factors such as the type of user relationship, i.e., weak connections or strong ties, the amount of information, and how much the information is spread. The new research should recognize and reduce new security and privacy risks; trust mechanisms and test recently built technologies and frameworks for enhanced privacy [110]–[113].
- **Coordinated inauthentic behavior (CIB):** With the 5G/6G network, cyber-attacks will become much more widespread. This botmaster's botnet may carry out different cyber-attacks, such as DDoS attacks, phishing, and spamming. Developing methods to detect coordinated and organized activity generates better outcomes than individual accounts' analysis. However, the main issues are the scalability of group-based detectors and inauthentic behavior. Hence, detecting such coordinated behaviors over the network is necessary to expose worldwide botnet breaches[66], [114]. It is a significant concern yet to be resolved, although with recent Quantum ML models, effectiveness and computation speed can be highly improved [115].
- **Blockchain-based decentralized OSN:** There is a lack of security mechanisms that firmly guarantee individuals' identity and the authenticity of system transactions to allow the possibility of executing major spam attacks on most networks [25]. The blockchain-based

framework can encrypt user information to make outcomes reliable and genuine. Blockchain technology and AI technology can create trust by verifying credible content spread via OSNs. OSN content can be powered and controlled by OSN users using a decentralized approach that prevents their content from being used for advertisements, predictions, and analyses by third-party apps, companies, etc. [116]. A decentralized blockchain-based distributed consensus mechanism can be used to control issues such as spying, snooping, spamming, account compromise, online fraud, drug trafficking, monitoring and detecting activities of cyber-criminals, etc. [39], [102], [117], [118].

1.4 Research Motivation

Most work in the field of spam detection is done on e-mail spam. E-mail servers maintain well-established spam filters such as blacklists, whitelists, greylists, and honeypot schemes for effectively detecting and classifying spam. However, traditional filtering techniques can't efficiently handle the social spam problem due to the large number of users and frequently changing strategies of spammers. Thus social spam detection in OSNs is still ongoing and is in the nascent stage.

With the advent of Web 2.0, spammers and spambots are predominantly more active in disseminating spam on OSNs due to their social nature and linked with many user accounts. Generally, spammers are responsible for triggering prominent issues, such as the privacy and security of audacious users, the contamination of search results with spam content, the consumption of server time, speed, storage, and computing power, and the degrading information quality. Recently, AI and DL-enabled Deepfakes have exacerbated misinformation at large.

Spammers use several evade strategies and Adversarial Machine Learning (AML) techniques to fool spam detectors. AML is an approach that cyber-criminals can use to trigger malfunctioning attacks on an ML/DL model that can be accomplished by engendering adversarial samples, i.e., altered feedback fed to the ML program to disorder its predictive potential. The existing ML/DL-based approaches have shown good results in the past few years; however, they still face many challenges, such as real-time data collection, class imbalance, feature engineering, multilingualism, spam drift problem,

unproductive, multi-modality, scalability, adversarial attacks, etc. Thus, detecting and eluding social spam content (text, images, and videos) and spammer accounts to counter these issues and their effect is highly relevant. Despite many advancements in the past few years, online users still face many security issues, threats, and challenges on OSNs due to unresolved prevalent spam. Hence, it is essential to build robust spam detector systems with better data pre-processing, feature extraction techniques, advanced pre-trained word embedding models, and a hybrid of better ML/DL approaches to efficiently detect spam content in OSNs.

1.5 Research Objectives

The objectives of the proposed research are as follows:

1. To study and investigate the Literature related to existing frameworks/approaches/techniques/models to detect spam in OSN.
2. To propose a framework for efficient spam detection in OSN.
3. To develop and verify the proposed framework for efficient Spam detection in OSN.
4. To validate the proposed framework w.r.t to existing framework/approaches.

1.6 Thesis Contribution

The following contributions are mentioned in this thesis to accomplish the aforementioned objectives:

To achieve the first objective, a comprehensive review of existing frameworks/approaches/techniques/models to detect spam in OSN has been done. Initially, the detailed background information related to social spam was explored extensively and presented as - the spamming process in OSNs, various types of accounts used by spammers to propagate and evolve online users to perform different types of cyber-attacks, the taxonomy of social spam, and social engineering attacks associated with social spam. Several machine learning and deep learning approaches have been reviewed for content spam detection and spammer account detection. Further, social spam content detection approaches are classified into text

spam and A.I. enabled multimedia spam (Fake images and videos) detection techniques, and the spammer account detection techniques are categorized into spammer accounts (Fake/Compromised and Cloned/Sybil account detection), spambot accounts, and spammer community detection. The tabular analysis provides information on datasets, features, feature extraction techniques, ML/DL techniques, and merits and demerits of the methods used for reviewed social spam detection approaches. Existing studies revealed that the performance comparison of such ML/DL approaches/models could be done using performance evaluation measures such as accuracy, precision, recall, F1-score, etc.

To fulfil the second objective, a framework for efficient spam detection in OSN is proposed with the ML and DL-based approaches for social spam detection. For ML-based social spam detection, voting ensemble models are proposed. Initially, the imbalanced datasets are balanced using NearMiss and SmoteTomek data-sampling techniques. The data pre-processing and feature generation is done using BoW and TF-IDF techniques before feeding into ML models. The PHVE model uses a hard voting mechanism, whereas the PSVE model uses a soft voting mechanism. Both PHVE and PSVE models, an ensemble of nine ML models, i.e., MNB, SVM, RF, XGB, ETC, BC, KNN, ADB, and GDB, are built with some parameter tuning. Later, the performance of the PHVE and PSVE models is validated over the testing datasets (20%). After the model evaluation on test data, the best parameter values are extracted, and the best prediction model is finalized by adjusting the parameters. In PHVE, the final class prediction is made using the majority voting of each classifier. However, in PSVE, the target label with the highest sum of weighted probability is chosen as the final prediction. Extensive ML-based experimentations are done on several imbalanced and balanced datasets to know performance improvement. Finally, compared to other ML baseline models, the performance of PHVE and PSVE models outperformed.

For DL-based approaches, i.e., GCB-SA and FCB-SA approaches are proposed and tested over balanced datasets. Initially, the input in the form of embeddings is generated using GloVe and FastText word embeddings over the balanced *combined dataset* and passed into the deep neural network comprised of Conv1D and Bi-directional recurrent neural network layers with the self-attention mechanism for improved context understanding and effective

results. The hybrid approaches for detecting social spam are examined on balanced social network data retrieved using the SmoteTomek data balancing technique.

For the third objective, the proposed framework with ML and DL-based approaches is developed for efficient spam detection in OSN. The proposed ML models (PHVE and PSVE) are trained and tested on imbalanced and balanced datasets. The balanced datasets are retrieved using NearMiss and SmoteTomek data balancing techniques. The DL-based hybrid approaches are evaluated over a balanced *combined dataset* built by merging the three benchmark datasets. All proposed ML/DL-based models/approaches are evaluated on defined experiment settings and optimized hyperparameters in terms of accuracy, recall, precision, F1 score, accuracy variation, ROC-AUC score, training/validation loss, training/validation accuracy, etc.

For the fourth objective, the proposed ML models and DL approaches are evaluated over several benchmark datasets. Later, the proposed ML models and DL approaches are validated by comparing the performance with existing frameworks/approaches/techniques. In future, the proposed ML/DL-based framework can be deployed on OSN websites/apps or in web browser extensions to detect text spam.

1.7 Thesis Organization

This thesis explores and proposes approaches for detecting spam in OSNs. The following outlines the information organized in the various chapters of this thesis:

Chapter 1: Introduction

This chapter presents the introduction to OSN and its misuse. The chapter provides background and important concepts related to social spam in OSNs to understand the thesis work better. The chapter mainly covers detailed information related to social spam, the spamming process in OSNs, various types of accounts used by spammers to propagate and evolve online users to perform different types of cyber-attacks, the taxonomy of social spam, and social engineering attacks associated with social spam. Besides, the chapter

highlights various challenges and open issues related to spam in OSN, research motivation, research objectives, thesis contribution, and thesis organization.

Chapter 2: Literature Review

This chapter provides a detailed Literature review analysis of recent ML/DL-based social spam detection approaches and spammer detection techniques. The social spam content detection approaches are categorized into text spam and AI-enabled multimedia spam (Fake images and videos) detection techniques. The spammer account detection techniques are classified into spammer accounts (Fake/Compromised and Cloned/Sybil account detection), spambot accounts, and spammer community detection. The summarized tabular analysis highlights datasets, features, feature extraction techniques, ML/DL techniques, performance assessment, and merits and demerits of the methods used for social spam detection. Finally, significant discussions and research gaps are summarized in the summary section.

Chapter 3: Proposed PHVE and PSVE models using machine learning models

This chapter investigates the class imbalance problem using various data sampling techniques. Several data balancing techniques, such as NearMiss, random oversampling, and SmoteTomek, are explored and implemented to solve the class imbalance problem. The ML models are fed with relevant features extracted using word representations techniques. A new *combined dataset* is built by merging three datasets. The chapter provides exploratory data analysis (EDA) on the *combined dataset* to find patterns, statistics, and graphical representations. The methodology of the proposed ML-based voting ensemble consists of seven parts: (i) Dataset acquisition, (ii) Data balancing, (iii) Data pre-processing, (iv) Exploratory data analysis, (v) word representation techniques, and (vi) ML-based voting ensemble models (PHVE and PSVE). The chapter provides an in-depth evaluation of the proposed hard voting ensemble (PHVE) and soft voting ensemble model (PSVE) for social spam detection over imbalanced and balanced datasets. Extensive comparison analysis reveals that the PHVE and PSVE models perform better on balanced datasets retrieved using SmoteTomek data-sampling technique. The chapter also

provides a performance comparison of PHVE and PSVE models with existing ML models/approaches.

Chapter 4: Proposed GCB-SA and FCB-SA using deep learning models

This chapter examines recent pre-trained word embedding techniques such as GloVe and FastText and NLP-based DL models such as CNN, RNNs, and self-attention mechanisms. The embeddings generated using GloVe and FastText pre-trained word embeddings over the balanced *combined dataset* are passed into the proposed hybrid deep neural network models comprised of Conv1D and Bi-directional recurrent neural network layers with the self-attention mechanism for improved contextual understanding and effective results. This chapter provides an in-depth evaluation of hybrid DL-based proposed GCB-SA and FCB-SA approaches. Experiments and comparisons with other DL approaches show that the proposed hybrid framework with deep learning-based approaches achieves better performance.

Chapter 5: Conclusion and Future Scope

In this Chapter, the conclusion of this thesis is presented. The proposed hybrid ensemble framework and approaches for social spam detection are concluded in this Chapter. This chapter also highlights some significant challenges and open issues related to social spam detection to benefit the research fraternity for future investigations.

Chapter 2 : Literature Review

This chapter examines social spam content and spammer account detection techniques/approaches. For a detailed Literature review, various reputed conferences, journals, and cybersecurity reports from multiple databases are explored specifically for spam detection in OSNs published mainly from 2015 to 2022. The Literature review is organized according to the taxonomy presented in Figure 2.1.

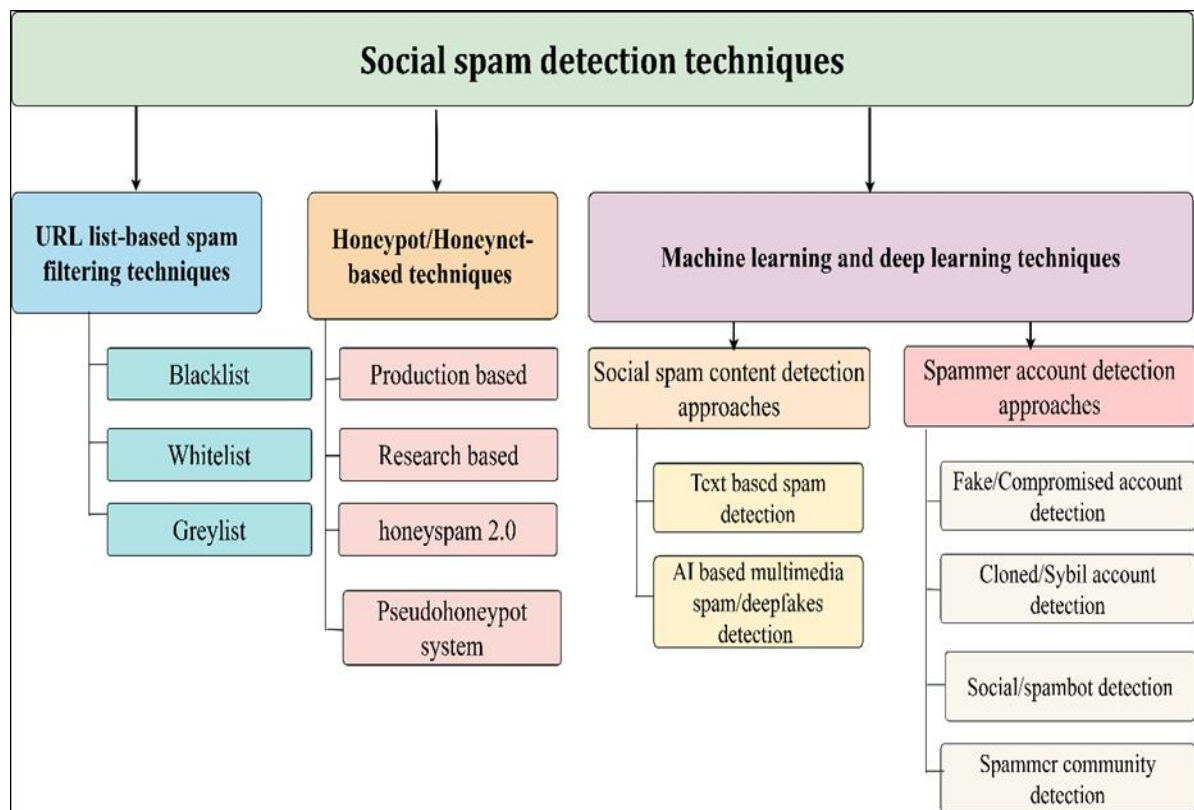


Figure 2.1: Taxonomy of various social spam detection techniques and approaches

Some work/contents of this chapter have been peer-reviewed and published in:

- Rao, S., Verma, A. K., & Bhatia, T. (2021). A review on social spam detection: challenges, open issues, and future directions. Expert Systems with Applications, Elsevier, 186, 115742. [I.F. 8.665]. SCIE.

After thorough analysis, the relevant information related to various dimensionality techniques, feature analyses, datasets used, and different traditional and ML/DL-based frameworks/techniques/approaches used, along with their merits and demerits, are extracted and summarized in tables. Deepfakes, a new kind of AI-enabled spam, are also investigated. Social spam detection may be addressed in a variety of ways. However, the focus of this thesis is predominantly on content spam. The chapter also explores various word representation techniques and contemporary pre-trained word embedding techniques. The chapter also describes the various performance evaluation measures used for evaluating ML/DL-based models/approaches for social spam. This chapter includes information on crucial challenges that must be addressed to construct effective spam detection systems. Subsequent chapters investigate and resolve several pressing issues.

2.1 Traditional Social Spam Detection Techniques

The researchers have introduced several counter-spam methods, such as URL list-based filtering methods, honeynet/pot-based methods, etc. The following provides a review of traditional techniques/approaches used for spam detection:

2.1.1 URL List-Based Spam Filtering Techniques

Frequently maintaining and analyzing URL lists using various methods such as whitelist, greylis, and blacklist are traditional operative approaches for spam classification.

Whitelist filters allow only trusted users list as senders list. The users that are not on the list are automatically blocked. In the greylis method, the server initially rejects and sends a failure message to the initiating server while receiving a message from an unknown user. If the server receives the message again, it is treated as non-spam and sent to the user, and his IP address is added to the allowed sender list or whitelist. Such methods are inconvenient if users expect time-sensitive messages from some unknown users and hamper the timeliness of messages. Blacklists are a list of spam URLs or IP addresses released and maintained by third-party organizations used as a filter to catch new spam. Google's safe browsing Application Programming Interface (API) has been used to avoid malicious links by scanning the URL among the list and blocking it before reaching the

victim [119].

The Blacklist has some limitations, such as it generates false positives in-case spammers use real IP addresses as a channel for spreading spam [120]. It requires manual labeling that makes it difficult to maintain the list if the number of spamming URLs increases with time and tiny URLs decrease the performance of blacklist methods. Also, blacklist methods are time-consuming and less feasible if spammers frequently change the features and spam content appearance. However, ML and DL-based approaches have improved the precision, detection speed, and functioning of URL list-based filters [121].

2.1.2 Honeypot and HoneyNet-Based Spam Detection Techniques

A honeypot security mechanism tracks, monitors, and detect intruders' activities in the information system. A honeynet is a network of highly interactive honeypots with firewalls and Intrusion detection systems (IDS) to record in-depth information. Spam profiles send a friend request to a honeypot without knowing it's a trap or a legitimate user profile. The honeypot records the information related to the spam profile and the timestamp on receiving the friend request, and later the friend request is rejected to get evaded from service providers [122], [123].

Honeypots can be categorized into production-based and research-based honeypots. Companies use production-based honeypots to record only partial information, whereas research-based honeypots are used to research threats and to provide security from such risks. Honeypots are used by governments, military, and research organizations to add cyber-criminals information but are challenging to deploy. Some researchers have used these to detect spam and discovered how malware disables itself to evade detection in honeypot environments [124].

Hayati et al. [125] built HoneySpam 2.0 spambot software to record web social spambot behaviors. Later collected information can detect spambots and filter spam content. Stringhini et al. [126] implemented honeypot approaches with honey profiles to identify spammers' behavior and features. Lee et al. [123] have used the honey-profile system on Twitter and MySpace to capture spam profile features during suspicious behaviors.

Eventually, they used ML classifiers to evaluate old and new spammer characteristics.

Paradise et al. [127] built a framework to control the honeypot network in OSNs, to detect advanced persistent threats (APTs) to organizations using honeypot profiles to monitor malicious friends' requests and messages to get initial information for a potential attack. Eventually, employees' actions towards such intrusions determine the success and attractiveness of produced honey profiles.

Cornelissen et al. [128] have developed a system to capture low-quality consumers who are highly influential from a South African political perspective. In their approach, the regular users fascinated by honeypots were analyzed on account of average age and verification.

Zhang et al. [129] implemented a Pseudo-Honeypot spammer detection system to analyze users' posts, messages, and interaction behaviors over social networks. In their ML-based framework, authors addressed shortcomings of conventional honeypot systems such as dataset collection and portability, attribute inconsistency, adaptability, faster performance, capability, etc.

2.2 Machine Learning and Deep Learning Approaches used for Spam Detection

The traditional approaches cannot perform well individually due to their drawbacks. In URL filtering methods, the cumulative list of malicious URLs, the legal IP address of spammers and short/tiny URLs, etc., made them cumbersome and time-consuming. On the other hand, honeynet/pot-based methods have challenges such as data processing, portability, scalability, adaptability, etc. By integrating advanced ML and DL techniques, the performance of the spam detection system can be enhanced. ML is a subfield of AI that allows learning without being directly programmed from the data. The general methodology used for social spam/spammer detection using ML/DL techniques is shown in Figure 2.2.

The ML and DL-based social spam detection involve the following steps:

- **Data collection:** The OSN datasets can be graph-based and non-graph-based. The dataset built from social network graphs is known as the graph-based dataset. In contrast, the dataset created from the features extracted from OSN using APIs provided by social networks is known as a non-graph dataset. Primarily, OSN data is gathered using APIs provided by OSNs. For example, Facebook has Facebook APIs, Sina-Weibo has Sina-Weibo APIs, and Twitter has Streaming, Rest APIs, and Twitter4j APIs.

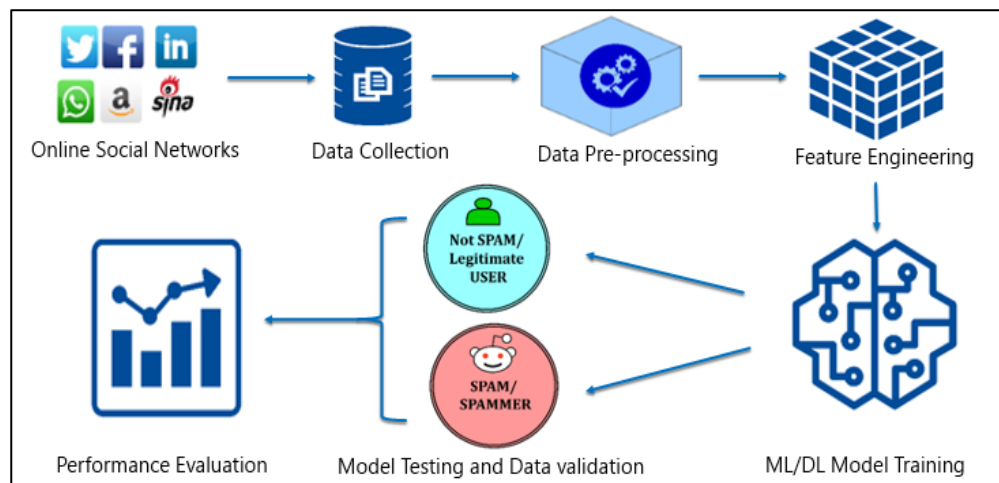


Figure 2.2: General ML/DL-based methodology used for social spam detection

- **Data pre-processing:** The next step after collecting the data from social networks is pre-processing the data. The collected raw data must be cleaned using pre-processing techniques before transferring to some ML/DL model. There are various phases in text data pre-processing, such as stop words removal, stemming, tokenization and segmentation, normalization and noise removal, etc.
- **Feature engineering:** The input to the model is given in a matrix formed from the best set of features using feature extraction and selection techniques. Feature extraction and selection techniques are part of dimensionality reduction to select dominant feature space. Feature selection is identified as a major concept to increase the effectiveness and efficiency of the ML model [130]. It specifies a subset of features from the full feature set without transforming them to build an

ML model. It reduces model complexity by removing unnecessary features and helps to accelerate the ML model training process. The reduced features help to reduce the overfitting problem and improve the overall accuracy of the ML model. Some commonly used feature selection methods are Chi-Squared (ChiSqr), Information Gain (Infogain), Particle search optimization (PSO), Ant colony optimization (ACO), Whale optimization algorithm (WOA), etc.

On the other hand, feature extraction converts the existing features into lower-dimensional space to reduce the overall computational cost or training time and to improve the ML model accuracy. It decreases the number of features in a dataset by creating new ones from the originals. Therefore, the new reduced set of features broadly summarizes the original collection of characteristics. Some of the popular feature extraction techniques used are principal component analysis (PCA), term frequency/inverse document frequency (TF-IDF), Word2Vec, and Bag of words (BoW), etc. In Table 2.1, dimensionality reduction techniques used to extract and select social spam/spammer detection features are mentioned. Several features have been studied from the reviewed ML and DL Literature and broadly categorized as content-based, user/profile and behavioral analysis, social network analysis, topological/network/graph-based, user account and behavioral analysis, statistical analysis, temporal analysis, physiological analysis. In Table 2.2, a summary of various types of feature analysis along with their feature attributes for social spam and spammer account detection in OSN is mentioned.

- **Model training and testing:** The ML and DL models are trained on selected features and tested on test-set, accompanied by performance evaluations such as accuracy (A), precision (P), recall (R), F1-measure/score (F), MCC, ROC, AUC, etc. based on the confusion matrix mentioned in Figure 2.3 to obtain spam detection results. The confusion matrix provides information about positive and negative predictions regarding TP, TN, FP, and FN. TP cases are those when the predicted result and actual result are true. FP cases are those when predicted results in True and actual results in False. TN cases were those when both predicted and actual results were false. In FN cases, predicted results are false and actual results are true.

The description and formula for some frequently used performance measures are mentioned in Table 2.3.

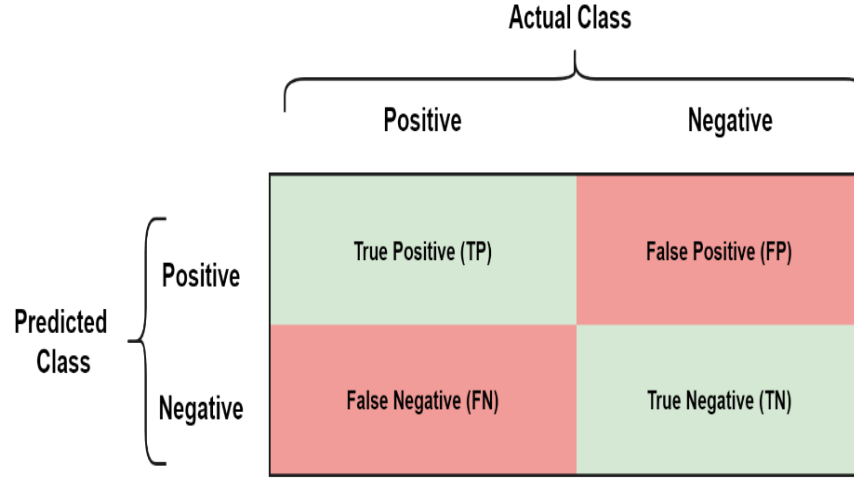


Figure 2.3: Confusion matrix

Table 2.1: Dimensionality reduction techniques used for social spam detection

Dimensionality reduction techniques		References
Feature extraction technique/(s)	TF, TF/IDF	[131]
	Snowball sampling	[132]
	PCA, Variational Autoencoders (VAEs), Time-Independent Component Analysis (TICA), and Long Short-Term Memory (LSTM) based Latent vectors	[84]
	Infogain, ChiSqr	[133]
Feature selection technique/(s)	RST algorithm, CFS, CON, COMM, Infogain, ChiSqr	[134]
	PSO, Simulated Annealing and ACO	[135]
	Correlation Based Feature Subset Selection (CBFS)	[40]
	PCA, IG, Gain ratio (GR), OneR, CFS, and DT-J48	[136]
	SVM -WOA ; SVM-PSO; SVM-GA	[137]
	Bi-directional-LSTM (Bi-LSTM) technique	[138]
	Genetic Algorithm (GA)	[139], [140]
	Stacked Denoising Autoencoders (SDAEs)	[141]

Table 2.2: Summary of various feature analyses used for social spam detection

Feature analyses	Feature type	Feature attributes	References
Content analysis	Text-based features	N-gram, Language model, LIWC, Message length, Message similarity, Word length, # of replies, # of messages/comments, spam words, abusive/aggressive words, sentiment score, semantic similarities, POS tags, etc.	[121], [131], [133]–[135], [138], [142]–[158]
	URL/Hashtag features	URL count, Blacklist, WHOIS, URL entropy, URL ratio, URL forwarding, Domain information, URL redirect, Lexical features, Descriptive features, etc.	[134], [135], [144], [148], [149], [155], [159]–[163]
	Image and video features	Image features, image frames, latent image features, eye blinking features	[96], [118], [164]–[167]
User/Profile and behavioral analysis	User-profile based features	Profile age, Profile picture, Profile description, user location, # of followers, # of following, Follower following ratio	[133], [134], [136], [139], [142], [144], [146], [151], [152], [154]–[156], [162], [168]–[173]
	Behavioral features	User profile metadata, Posting and clickstream behavior, retweeting, resharing, social behavior, etc.	[27], [29], [137], [141], [146], [151], [152], [156], [173]–[175]
Social network analysis	Topological/ Network-based features	Degree distribution, Centrality measure, Degree density, Foreign in/out probability, Clustering coefficient, Link analysis, Community structure, Community membership.	[132], [136], [145], [151], [152], [155], [161], [169], [171]–[173], [176]–[178]
Statistical analysis	Similarity, distance, distribution, correlation coefficient	Degree, relevancy of degree, distribution characteristic, clustering distribution, shortest path distribution characteristics, size distribution characteristics of connected groups, and sparsity distribution characteristics	[27], [29], [84], [136], [138], [150], [157], [176], [178]
Temporal analysis	Date, Time features	User activity time, sequential patterns	[84], [136], [151], [152], [161], [165],

			[166], [169]–[171], [179]–[182]
Physiological analysis	Eye blinking, Face-warping, Heartbeat	Face color, heart rate, eye-blinking rate	[165], [181], [183]– [185]

Table 2.3: Performance measures used for social spam detection

Performance measure	Description
Accuracy (A)	The ratio of corrected predictions over the overall number of predictions. $\text{Accuracy} = \frac{TP+TN}{(TP+FP+TN+FN)}$
Equal Error Rate (EER)	A statistical output parameter used for the biometric system gives the ROC Curve position, usually where False Acceptance Rate (FAR) and False Rejection Rate (FRR) are equal. For better accuracy of the system, the EER value should be low.
F1-Score/F1-Measure (F)	It is defined as the harmonic mean of precision and recall. $\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}$
False Negative Rate (FNR)	The ratio of positive samples that were incorrectly classified. $\text{FNR} = \frac{FN}{TP + FN}$
False Positive Rate (FPR)	The ratio of negative samples that were incorrectly classified $\text{FPR} = \frac{FP}{TN + FP}$
Matthews Correlation Coefficient (MCC)	MCC score varies between -1 to 1, where -1 specifies a fully wrong binary classifier, whereas 1 specifies an entirely correct binary classifier. $\text{MCC} = \frac{(TP * TN) - (FP * FN)}{\sqrt{[(TP + FP) * (FN + TN) * (FP + TN) * (TP + FN)]}}$
Precision (P)/Positive Predictive Value (PPV)	The ratio of true positives out of all detected positives. $\text{Precision} = \frac{TP}{TP + FP}$

Kappa Score (K)	It is calculated using observed accuracy and expected accuracy. Observed accuracy is the number of correctly classified instances, while any random classifier predicts the expected accuracy value [171], [175], [186].
	$\text{Kappa} = \frac{\text{observed accuracy} - \text{expected accuracy}}{1 - \text{expected accuracy}}$
Klout Score (KS)	A numerical value ranges between 1 to 100 is used to rate OSN users according to their social influence [168].
Recall/Sensitivity/True Positive Rate (TPR) / Hit Ratio (R)	The ratio of true positives that are correctly classified. $\text{Recall / Sensitivity / TPR / Hit Ratio} = \frac{\text{TP}}{\text{TP} + \text{FN}}$
Receiver Operating Characteristics (ROC) and Area under the ROC Curve (AUC)	It is a 2-D graph showing TPR as Y-axis and FPR as X-axis. AUC under ROC provides a total performance measure across all potential classification thresholds.
Specificity/True Negative Rate (TNR)	The ratio of true negatives over total negative predictions $\text{Specificity / TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$

The following subsections provide recent work done on social spam content and spammer account detection using ML/DL techniques:

2.2.1 Text Spam Detection Approaches

Text-based spam content includes malicious URLs, hashtags, fake reviews/comments, posts, SMS, chat messages, etc. In ML, spam is categorized as a predictive modeling problem where a class label is predicted for a given input dataset. The research community has worked with various ML techniques such as supervised, unsupervised, semi-supervised, Metaheuristic Optimization Algorithms (MOA), Ensemble Learning (EL), and DL methods to classify text-based spam. The research work carried out in the form of fake reviews, posts, and hashtags to classify text spam is described below:

The Supervised Learning (SL) methods for text spam detection work with labeled data. The most common classification methods used for spam detection are Decision Trees (DT),

Naïve Bayes (NB) classifier, Logistic Regression (LR), Random Forest (RF), Adaboost and Support Vector Machine (SVM), etc. [133], [168].

Liu et al. [155] have proposed an iterative algorithm (IA) to find spam posts and performed experiments on real-world datasets to identify spam hashtags in an OSN with collaborative knowledge that can be extended to detect spammers at the user level.

Narayan et al. [143] have used many SL algorithms such as DT, NB, SVM, KNN, RF, and LR on three features: LIWC, n-grams, and Part of Speech (POS) tags with a sentiment score to detect review spams. The features were tested individually and combined with many SL algorithms; eventually, the LR classifier technique outperformed when tested by combining unigram and Linguistic Inquiry and Word Count (LIWC) with a sentiment score.

Gupta et al. [142] proposed an ML-based framework that applied SVM, Neural Network (NN), RF, and Gradient Boosted (GB) algorithms on features from the user and tweet text to detect spam on tweets. The tweet text feature identifies spam tweets, spammers, or malicious user accounts. The experiment was performed on 400,000 public tweets and extracted 30 words to provide maximum information gain to classify tweets with 91.65% accuracy.

Mehmood et al. [158] have proposed a prediction model for spam comments based on EL. It took TF, TF/IDF text features using stacking as inputs to RF and Gradient Boosted Trees (GBT), and outputs of these learners are given to DT for training the final model. When compared, their model showed 92.19% accuracy.

Saumya & Singh [157] listed various drawbacks of SL methods, such as the manual labeling of online reviews on product quality to categorize spam and not spam, data imbalance problems, and costly computations for analyzing similarities among reviews. Their proposed system used three different classifiers, i.e., GB, RF, and SVM, to work with new features such as sentiments of reviews and comments using a sentiment mining approach with a sentiment variation threshold of 0.6, deviation in rating, and content-based factor. The experiment was done on 1332 reviews from Amazon that resulted in a 91% F-

score. The SL methods are less productive in real-time scenarios due to spammers' varying nature and class imbalance problems, where labeled spam accounts are more than non-spammer accounts. Moreover, the SL approaches are less efficient in real-time scenarios due to the spam drift concerns and class imbalance problem where branded spam accounts are more than non-spammer accounts.

Ahmed et al. [131] have mentioned that opinion spam spreads false information or theories in fake news and reviews. They also stated that such misinformation has daring consequences for online users, and it's difficult to distinguish between real and fake reviews/news. Their implementation performed n-gram analysis with two feature extraction techniques, i.e., TF and TF/IDF, with six ML classification techniques, i.e., Stochastic Gradient Descent (SGD), SVM, Linear Support Vector Machine (LSVM), KNN, LR, and DT. The classification techniques were tested over public datasets and the fake-news dataset. The accuracy of classifiers improved as the number of features increased, and accuracy declined as the size of the n-gram increased.

The Unsupervised Learning (USL) algorithms are used when the data used to train is neither classified nor labeled. The USL methods group the data into various classes depending on their characteristics. The learning is done by detecting the similarities among various dataset instances. Clustering techniques are used to identify spam campaigns.

Sohrabi & Karimi [135] have investigated Facebook wall post comments for spam detection using USL techniques such as hierarchical, partition-based, PCA-based, stream-based, DB index, pairwise similarity, etc. The unsupervised techniques are also incompetent due to ample data storage and less efficient clustering algorithms to detect spam campaigns in large OSNs.

Dutta et al. [134] proposed an RST attribute selection algorithm for finding smaller subsets of attributes selected for better spam classification after comparing the proposed methodology with existing methodologies for attribute selection, such as Rough Set Theory (RST) algorithm, Correlation-based Feature Selection (CFS), Consistency Subset Evaluation (CON), Community detection (COMM), Infogain, and ChiSqr over five different OSN spam datasets. They found their method selected smaller subset attributes

with better classification performance.

Tajalizadeh & Boostani [153] have clustered Twitter spam messages using the DenStream clustering method with an incremental Nave Bayes classifier (INB) on micro-clusters to capture the mean and border of micro-clusters instead of the Euclidean distance approach. Their INB-DenStream technique outperformed previous known stream clustering methods such as CluStream, DenStream, and StreamKM++ on Twitter datasets in terms of accuracy, precision, recall, and reduced complexity.

Xia [154] has addressed anti-spam filter performance issues owing to incremental obscured words and frequent rule updates. Besides, they developed a rule-based system (RBS) using the Hash Forest algorithm and rule encoding technique to filter spam messages from bulk SMS messages with $O(1)$ time complexity.

The Semi-Supervised Learning (SSL) techniques are useful for both labeled and unlabeled data. Classification models are combined with unsupervised techniques to train the labeled and unlabeled data. Some SSL techniques for detecting spam are co-training combined with k-medoids [156], Transductive Support Vector Machine (TSVM), and self-training. SSL techniques show a better result in the scarcity of publicly labeled data to identify malevolent accounts [30].

Sedhai & Sun [144] implemented a Semi-Supervised Spam Detection (S^3D) approach for detecting spam that consist of one module for spam detection and another for model update. The first module worked in real-time mode, and the other worked in batch mode. The first module consists of four lightweight detectors, i.e., blacklisted domain, near-duplicate, reliable ham, and multi-classifier-based detectors to detect spam tweets. The batch module updates the information needed by the spam detection module from time to time. Features used are obtained from the user account, historic tweets, and social graph. Their proposed approach robustly learned new spam event trends and provided accurate results for spam detection on Twitter.

Singh & Batra [162] proposed a semi-supervised ensemble framework based on Probabilistic Data Structures (PDS) as classifiers to detect tweets as ham or spam on

Twitter. The PDS uses Quotient Filter (QF) for querying URLs, spammers, spam, and Locality Sensitive Hashing (LSH) for pattern search with less computational effort. Furthermore, the proposed system was tested and evaluated on F-score, precision, and recall.

Yilmaz & Durahim [145] developed a semi-supervised Spam Review Representation (SPR2EP) framework with integrated feature vectors extracted from review text and reviewer-product networks. Feature vectors have been learned using document learning and node embedding algorithms, and later inputs were fed into an ML-based classifier to detect opinion spam. The performance of SPR2EP has outperformed when compared with other existing techniques on three different datasets YELPChi [187], YELPZip, and YELPNYC [188].

ML methods were ineffective and slow-moving in the case of high-dimensional data and the varying nature of spammers [121], [146]. For the fast detection of spam over OSNs with a massive amount of data, it becomes essential to explore DL methods with efficient feature selection mechanisms [147]. The recent advancements in computing power in terms of capable hardware, software, and Graphical Processing Units (GPUs) helped solve complex problems with massive data in less time. DL methods are better for extracting features from text, images, video, and audio. DL techniques, such as belief networks, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Deep Neural Networks (DNN), etc. have been used for social network analysis (SNA), bot behavior detection, Natural language processing (NLP), computer vision, recommender systems, medical image analysis, drug discovery, fraud detection, bioinformatics, image processing, advertisement click fraud detection, etc. [56].

Wu et al. [121] have developed a DL technique to identify spam on Twitter by learning the syntax of many tweets using the word vector technique to perform pre-processing and create high-dimension vectors. Their DL-based MLP classifiers on the real Twitter dataset consisting of labeled data with 376206 spam and 73,836 non-spam tweets collected over 600 million real-world tweets. Their proposed MLP technique outperformed when compared with other text-based classifiers (Palladian, Complementary Naive Bayes, and

Complementary Naive Bayes (Frequencies)) and feature-based ML methods, i.e., NB, RF, and DT (C4.5).

Jain et al. [147] have implemented a Semantic Convolutional Neural Network (SCNN) model that uses a CNN technique with a semantic layer. The Semantic layer used an NLP technique called Word2Vec to map the word with word vectors from Google's Word2Vec. The WordNet and ConceptNet search for similar words for a given word. The model is implemented over two datasets, i.e., a Twitter dataset with 94.40% accuracy and 86.5% accuracy on the SMS spam dataset.

Feng et al. [148] have proposed a DL-based multistage and elastic framework over Mobile OSNs. Their system uses a mobile terminal to identify messages in URLs and QR codes etc. These messages are then directed to the server. The server uses a detection queue that elastically discovers messages with limited resources. The framework was found to be accurate and efficient. The system provided a better utilization of resources with a low false-positive rate on the Sina-Weibo dataset built using Sina-Weibo API.

Selvaganapathy et al. [149] detected and classified malicious URLs using a deep neural network technique and a feature selection Boltzmann machine. The results of the Deep Belief Network (DBN) model are compared with SVM, ANN, and NB on a dataset of 27700 URL samples built from existing datasets such as the spambase dataset [189], phishing dataset [190], and malware domain list [191]. The DBN model provided a better result with fewer false positives. The work can be extended by implementing other DL methods such as CNN, Multi-label classification algorithms such as Multi-Label k-Nearest Neighbors (MLkNN), Random K-label subsets (RAKEL), and Back-Propagation Multi-Label Learning (BP-MLL) for malicious URL detection.

Thejas et al. [56] built a DL-based hybrid model for click fraud detection. The model involved an Artificial Neural Network (ANN) and autoencoders. ANN is used to gain learning and pass knowledge to the other layers in the neural network, and auto-encoders are used to acquire the distribution of human clicks. The model provided high accuracy when tested on a real-time ad-clicks dataset.

Singh et al. [150] have classified aggressive behavior on social networks using DL techniques. The CNN model is used on the TRAC-1 (Trolling, Aggression, and Cyberbullying) dataset consisting of posts in the Hindi-English language and categorizes each post/comment into three groups, i.e., overtly aggressive and covertly aggressive, and non-aggressive. The model predicted 73.2 accuracy.

Ban et al. [138] have automatically extracted Deep Learnt Features from Twitter text using the DL-based Bi-LSTM technique for spam detection. They trained different ML models on deep-learned features, word2vec features, and statistical features and found models trained on deep-learned features more accurate with higher F1-Scores.

The feature selection using optimization algorithms helped ML models to get acquainted with only optimized features. Sohrabi & Karimi [135] designed an efficient spam comment filtering system using PSO, simulated annealing, and ACO as feature selection methods for analyzing Facebook wall post comments. These techniques are combined with SVM and DB index clustering techniques to detect spam more quickly and precisely. Al-Zoubi et al. [137] proposed a hybrid ML model using the SVM and Whale Optimization Algorithm (WOA) over four different lingual datasets (English, Spanish, Arabic, and Korean) collected from Twitter for detecting spam. The model extracted content, behavior, and characteristic features. Their model outperformed in terms of recall, F-score, AUC, accuracy, and positive predictive value (PPV) than several other existing algorithms. The summary of text spam detection approaches used in OSNs is given in Table 2.4.

Table 2.4: Summary of text spam detection approaches used in OSNs

Reference	Dataset used	Features used	ML/DL approach used	Techniques/methodology used	Merits	Demerits
[155]	Delicious dataset - http://delicious.com	Topological, semantic, and profile based	IA	Iterative algorithm with collaborative knowledge.	Used collaborative knowledge	Personomies not taken into account
[121]	Twitter dataset	Text features	SL, DL	MLP Classifier with Word2Vec technique; RF, Palladian, complimentary NB, complementary NB (Frequencies), and DT (C4.5)	MLP classifier outperformed when one OSN dataset compared with basic ML classifiers	The framework is only tested on
[131]	Dataset 1 [192], dataset 2 from reuters.com and kaggle.com	n-gram features	SL, MOA	SGD, SVM, LSVM, KNN, LR, and DT.	Better performance for Fake news data	Lacks semantic, statistical, and writer's style features
[157]	Product reviews dataset from Amazon.in	Helpful votes, rating deviation, review sentiment, average cosine similarity, comment sentiment, content-based factor, number of comment	SL, EL	Sentiment mining approach using RF, GB, SVM ML classifiers	Automatic labeling of reviews	Tested only on one category of electronic products on Amazon only.

[134]	Dataset 1[193], dataset 2 [194], dataset 3 [195], dataset 4 [196], dataset 5 using Twitter API	Text-based, URL-based, SL, EL profile-based		Attribute selection method: Rough set theory-based algorithm; ML classifiers: NB, RBF network, LR, LogitBoost, decision tree, OneR, and Random tree.	Proposed attribute selection methods performed better.	Used only discrete/categorical attributes; Work can be extended with continuous attributes.
[158]	YouTube Spam collection dataset [197]	TF/IDF features	SL, EL	RF, GBT, and DT algorithms	Worked on the latent semantic meaning of comments	The framework can be tested on other spam datasets.
[142]	Hspam14 dataset [113]	User-based, tweet based, and text-based	SL, EL	SVM, NN, RF, and GB algorithms	Used bag-of-words model to detect spam words	Work can be extended using a self-learning algorithm.
[143]	Opinion spam datasets [192], [198]	LIWC, POS Tags, N- Gram feature, sentiment score	SL, EL	DT, NB, SVM, KNN, RF, and LR	Classified spammers on positive and negative reviews	Work can be extended using USL and SSL techniques.
[144]	HSpam14 [113]	Hashtag, Content, User, SSL, SL and domain-based features		S^3D and S^3D model with update module; NB, LR, RF	Captured new spamming patterns successfully	The framework can be integrated with user-level spam detection
[162]	Dataset generated using python script	Spam URLs, Spammy words, spam users,	SSL	PDS, LSH, Majority Voting, and Markov-based updation model	Ensemble-based framework	It can be tested on other social platforms

		ham/spam tweets				
[135]	Sina-Weibo dataset [133]	Hash-Tag, reply, #Like, spam words, URLs, #shares, burst time, friends, followers, message size	SL, USL, MOA	SVM, DT; Unsupervised technique used: DB index; PSO, ACO	Combined supervised and unsupervised methods to improve the precision	The system can be tested with other SL and USL techniques.
[145]	YELPChi [188]; YELPNYC, YELPZip [187]	Review text-based, Reviewer, and product network-based features	SSL	SPR2EP framework	Combined Node2vec and Doc2vec	DL methods can be incorporated and tested.
[147]	SMS spam dataset [199]; Twitter dataset using Twitter API	Text features	SL, DL, EL	SCNN: CNN and semantic layer (Word2Vec, WordNet, and ConceptNet); SVM, NB, ANN, RF, KNN	Performed automatic feature learning using a semantic layer	Worked on a small dataset of one language only
[148]	Sina-Weibo dataset using Sina-Weibo API	Text messages and malicious URL	SL, DL, EL	Multistage framework with word2vec model; LR, RF, GBT, and NB; DL	Accurate and efficient framework	Worked on a small dataset
[149]	Malicious URLs dataset	Spam URL, Phishing URL, malware URL features	DL, SL	DBN Model; ANN, SVM, NB	Fast and reduced false positive	The detection rate can be improved using normalization. The model was tested on small datasets only.
[137]	Used datasets in Arabic, English, Spanish,	Content-based, behavior-based and	SL, MOA	SVM -WOA ; SVM-PSO; SVM-GA	Identified the most prominent features	It can be tested with other Classification and optimization

	Korean, and multilingual context.	characteristic-based		ML classifiers: J48, k -NN, MLP, NB, RF, and SVM-Grid search	in different linguistic contexts	techniques.
[56]	Ad-clicks dataset [200]	Device IP and ID features	DL	ANN; unsupervised auto-encoder; semi-supervised GAN	Accurate and efficient framework	An imbalanced dataset with fewer amount of bot clicks
[150]	TRAC-1 dataset from Hindi-English code [201]	Text-based, abusive, or aggressive words, numerical features	SL, DL	MNB, DT, SVM, MLPs, LSTM, CNN	Worked on mixed Hindi/English sentences from social media	Used small dataset
[138]	Twitter dataset using Twitter Streaming API	Twitter-text, Word2vec, statistical features	SL, DL, EL	Bi-LSTM technique; ML Models: DTs- (C4.5, J48), RF, KNN, NB, SVM, and NN	Automatic feature generation	Framework tested only on the labeled dataset. Difficult to label a large number of spam tweets.
[153]	Low-dimensional Twitter datasets	Tweet-based and user-based features	USL	INB-DenStream; CluStream, DenStream, and StreamKM++	The INB-DenStream framework offered low complexity and better performance	The framework can be extended with an effective real-time distance learning approach.
[154]	SMS messages	Text-based	USL	Hash forest	Maintained constant time-complexity	Rule-based filtering algorithms can be extended with incremental rules, on-pass compressing, and rough set techniques.

2.2.2 Word Representation Techniques used for Text Spam Detection

As classification algorithms cannot work directly with textual data, the next step is to extract key features from textual data by converting the data into numeric representations. The most common word representation techniques are one-hot encoding, Count vectorizer (CV), and Hash vectorizer (HV). The description for the frequency-based vectorization techniques is as follows:

Bag of Words (BoW): In BoW, a sentence/document is characterized as a multiset of words with the frequency of each word that appeared in the document or sentence. The BoW excludes information such as phrases, syntax, and word order [142], [146]. BoW, CountVectorizer (CV), and TF-IDF use the TfidfVectorizer of the sklearn library for frequency-based vectorization.

Term Frequency-Inverse Document Frequency (TF-IDF): It is an improvement over BoW as it multiplies the frequency of words within a document by the rarity with which they appear in the entire corpus. The TF denotes the frequency with which a word appears in a sentence concerning all other words, and the IDF denotes the downscaling of words often appearing in other comments/documents. Words with a higher frequency will have a higher weight, whereas words with a low frequency will have a lower weight [158], [202]. The TF-IDF weight of a word $w_{t,d}$ in a document/corpus can be calculated using Eq. (1):

$$w_{t,d} = tf_{t,d} \times \log\left(\frac{N}{df_t}\right) \quad (1)$$

where t is term/word, d is document, $tf_{t,d}$ is number of occurrences of t in d , df_t is number of documents containing t , N provides total number of documents in the input corpus.

The word representation techniques such as BoW and TF-IDF create sparse and high-dimensional feature vector space that makes training ML/DL models hard. Also, BoW and TF-IDF can't hold the word's semantic meaning. This problem can be solved using dense and low-dimensional feature representations using neural network-based pre-trained word

embeddings. Besides, pre-trained word embeddings effectively identify a language's linguistic (syntactic and semantic) regularities [203].

2.2.3 Pre-trained Word Embedding Techniques

The feature extraction techniques retrieve essential features from the textual data. In comparison, traditional word representation is helpful for long text messages and documents and not for short text messages such as SMS, tweets, comments, news headlines, etc. Moreover, traditional techniques wouldn't improve the classification results and failed to capture the contextual relationship between the words. These issues are solved by word embedding, which describes words using dense vector representations where similar words have similar vectors. It also helps in catching semantic similarities. Word embedding maps each word or phrase from the vocabulary to an N-dimensional vector of real numbers. Its goal is to collect semantics and syntax on low-dimensional vectors. Several word embedding approaches have been employed to convert textual words into integers, known as word vectors, as input to ML models. Word embedding is trained from a large corpus. Generally, language or domain-specific word embeddings capture the statistical relations of all words in the corpus [204]. Employing existing pre-trained word embeddings is more viable than training them. Some of the recent pre-trained word embedding techniques are described as follows:

Word2Vec: Word2Vec technique uses predictive analysis to bring context based on the hypothesis that similar words have similar dimensions [205]. It takes input as a text corpus to create a vector representation for each word using the Gensim Python library. Word2Vec predicts a word's co-occurrence with its neighbors. Word2Vec creates a high dimensional vector for each word using shallow neural networks with two hidden layers, i.e., Continuous bag of words (CBOW) and the Skip-gram model [147], [148]. The CBOW model architecture predicts the current target word 'w' from the source context words (neighboring words). The CBOW model objective is to maximize the log of the probability in Eq. (2), where w_t represents the current word.

$$J = -\log \hat{P}(w_t | w_{t-n+1} \dots w_{t-1}) \quad (2)$$

Skip-gram predicts the neighbors of a word ‘w’ in a sentence or a document. The model aims to maximize the average log probability (Eq. (3)) in a given set of training words $w_1, w_2, w_3 \dots w_T$, where ‘n’ is the training context size that can be a function of the center word w_t . A higher ‘n’ value means more training examples and better accuracy at the cost of more training time [206].

$$J = \frac{1}{T} \sum_{t=1}^T \sum_{-n \leq j \leq n} \log p(w_{t+j} | w_t) \quad (3)$$

The pictorial representation of CBOW and the Skip-gram model of the Word2Vec technique is shown in Figure 2.4.

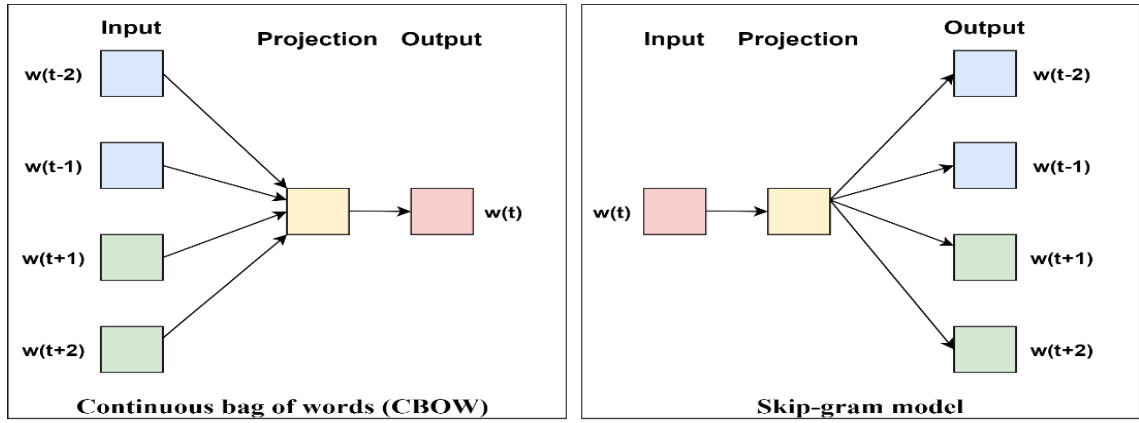


Figure 2.4: Word2Vec word embedding with a CBOW and skip-gram model

Word2Vec is a window-based model that limits the number of words using a window size parameter in each context. It is ineffective in long documents and fails to catch sub-word information [138], [202]. It can only produce vectors from local vocabulary and cannot handle words not in the vocabulary. Unambiguity, or words with numerous meanings based on context, is another issue with word2Vec [207].

GloVe: Global vectors were created at Stanford University. It's a count-based unsupervised learning approach for extracting word representations from a corpus by aggregating global word co-occurrence matrices [208]. The GloVe model's central idea is that word-to-word co-occurrence probability ratios can encode meaning. For this, consider a function F that models the ratio of probabilities relationship as mentioned in Eq. (4)

$$F(w_x, w_y, \tilde{w}_z) = \frac{P_{xz}}{P_{yz}} \quad (4)$$

where w_x, w_y are the words in context, $\tilde{w}_z$ are the words out of context, P_{xz}/P_{yz} depends on three words x, y , and z

The GloVe is better than Word2vec because the Word2Vec model is prediction based and captures the relations between words in a window. For the DL-based experimentation, the GloVe embedding '*glove.6B.300d.txt*' with 1.04GB in size is used. The GloVe embedding has 300 dimension word vectors trained on 6 billion words with 400k dictionary words [208].

FastText: FastText is a library developed by Facebook in 2016 as an extension of the Word2Vec approach for efficient learning of word representation and sentence classification [209]. FastText divides words into n-grams rather than providing individual words to the Neural Network (sub-words). FastText model training takes longer than other word embeddings since n-grams exceed the number of words. It outperforms Word2Vec and correctly represents rare words. Therefore, if a word was not seen during training, its embeddings can be determined by splitting the word into n-grams [204]. Word2Vec and GloVe cannot generate vector representations for words not included in the model dictionary, but this is a significant advantage of FastText. Moreover, FastText is fast, but the limitation is its huge size.

For DL-based experimentation, the FastText embedding (*wiki-news-300d-1M-subword.vec*) having 2.04GB size is used. FastText has 300 dimension word vectors trained with subwords from Wikipedia 2017, UMBC webbase, and statmt.org news dataset [210]. The comparison of various word representation and word embedding techniques is given in Table 2.5.

Table 2.5: Comparison of word representation and word embedding techniques

Word representation techniques	Type	Merits	Demerits
BoW	Frequency-based	- Computation is easy. - Essential metric to extract terms. - Compatible with the unknown word.	- It can't capture the word's semantic, syntactic, and sentiment information.
TF-IDF	Frequency-based	- Computation is easy. - Less memory. - Ideal for text problems with larger words and document files. - Basic technique to mine the descriptive terms. - Similar terms do not influence results due to IDF.	- It can't capture the word's semantic, syntactic, and sentiment information.
Word2Vec (Non-contextual word representation) [205]	Prediction based	- Captures the semantics & syntactic information of words. - Pre-trained Word2Vec trained on a large corpus.	- It can't capture contextual information. - It can't capture out-of-vocabulary (OOV) words. - Requires a larger corpus to learn
GloVe (Non-contextual word representation) [208]	Frequency-based	- Use vectors to find sub-linear relationships - Lower weight will not impact the training of stopwords.	- It can't capture contextual information. - Requires more memory. - It can't capture OOV words. - Requires a larger corpus to learn.

FastText (Non-contextual word representation) [209]	Prediction based	- Fast. - More accurate in finding rare words. - Can handle OOV words.	- Huge Size - Requires longer training time. - It can't capture contextual information. - Requires more memory.
---	---------------------	---	--

2.2.4 AI-Enabled Multimedia Spam (Deepfakes) Detection Approaches

Deepfake is manipulated synthetic multimedia content created using DL techniques and fake content that has never really taken place. Deepfake technology uses GPUs, AI, facial mapping, and Neural Networks to learn from large sample datasets to simulate human behavior, voice, expressions, variations, etc. Deepfake content seems genuine as they use real images, backgrounds, sound effects, etc. Deepfake content created using advanced AI and DL techniques can modify the facts and eradicate trust by giving a forged reality [118]. Deepfake technology can be valued in applications such as movies, games, healthcare, education, industries, etc. Deepfake content brought many challenges that need to be addressed quickly. Deepfake synthetic media can be created with GANs on a sample dataset of images, videos, and sounds that work on two network generators and discriminators. In other words, using this technology, the faces of two persons can be swapped in a video [100]. Original videos are tampered with face images of famous people. A new arms race between Deepfake techniques and detection algorithms has probably begun.

Kumar et al. [164] have detected image spam by implementing DL-based CNN techniques over an image dataset with 810 natural and 928 spam images. The CNN model identified features automatically, provided 91.7% accuracy, and outperformed other ML techniques.

Korshunov & Marcel [96] have evaluated some face recognition approaches, such as lip-syncing, Visual Geometry Group (VGG), Facenet, Image Quality Metrics (IQM), and SVM, etc., on Deepfake videos created using GANs from the VidTIMIT database. After evaluation, the authors found these methods unproductive for Deepfake detection.

Guera & Delp [166] have evaluated a temporal-aware pipeline that used CNN, RNN, and LSTM-based systems over the HOHA video dataset to spot Deepfake videos. The CNN technique mines feature from several frames of images to train the RNN model. Then training data is fed into LSTM for building temporal sequence descriptors to classify between original and tampered videos.

Li & Lyu [183] concluded that existing Deepfake algorithms yield low-resolution images, which must be enhanced to correctly encompass the original faces in the source video, which leaves noticeable artifacts in the subsequent Deepfake videos. Their model aimed at affine face-warping objects to differentiate real and fake images with a dedicated Convolutional Neural Network (CNN) algorithm. Four versions of CNN, i.e., VGG16, ResNet50, ResNet101, and ResNet152, are trained using training data. The facial regions are initially detected and extracted using the dlib software package. Later faces are aligned into multiple scales and smoothed each scale by a Gaussian blur with kernel size (5×5). To simulate the objects in the Deepfake output pipeline, the smoothed face suffers an affine warp back to the same dimensions as the original faces. In processing training data, their model saves time and resources.

Nguyen et al. [97] surveyed several methods to create and detect Deepfake images or videos. Deepfake videos are generated primarily through FakeApp, developed by Reddit OSN, using DL-based autoencoders and decoders pairing strategy, as shown in Figure 2.5. In part 1 of Figure 2.5, the same autoencoder is used to mine latent features from two different facial images, A and B, later used by two decoders, i.e., decoder A and decoder B, for training and reconstructing facial images. In part 2 of Figure 2.5, latent features extracted by the common encoder for image A are decoded with decoder B to reconstruct image B from image A. Some other tools, such as Faceswap, DFaker, DeepFaceLab, and Tensorflow-based Deepfake-tf, are based on a similar pairing strategy followed by FakeAPP. Later in 2020, Nguyen & Derakhshani [211] applied four DL models (LightCNN, Resnet-50, DenseNet-121, and SqueezeNet) biometric comparison of eyebrows to distinguish the face swap. Eventually, they trained their models on the VISOB dataset and tested them on the Celeb-DF dataset. The eyebrow images are resized, rotated, and cropped to match the different DL models input requirements. The authors achieved

an Equal Error Rate (EER) of 20.7% and 0.879 AUC on Celeb-DF, significantly better than other recently reported results on the Celeb-DF dataset.

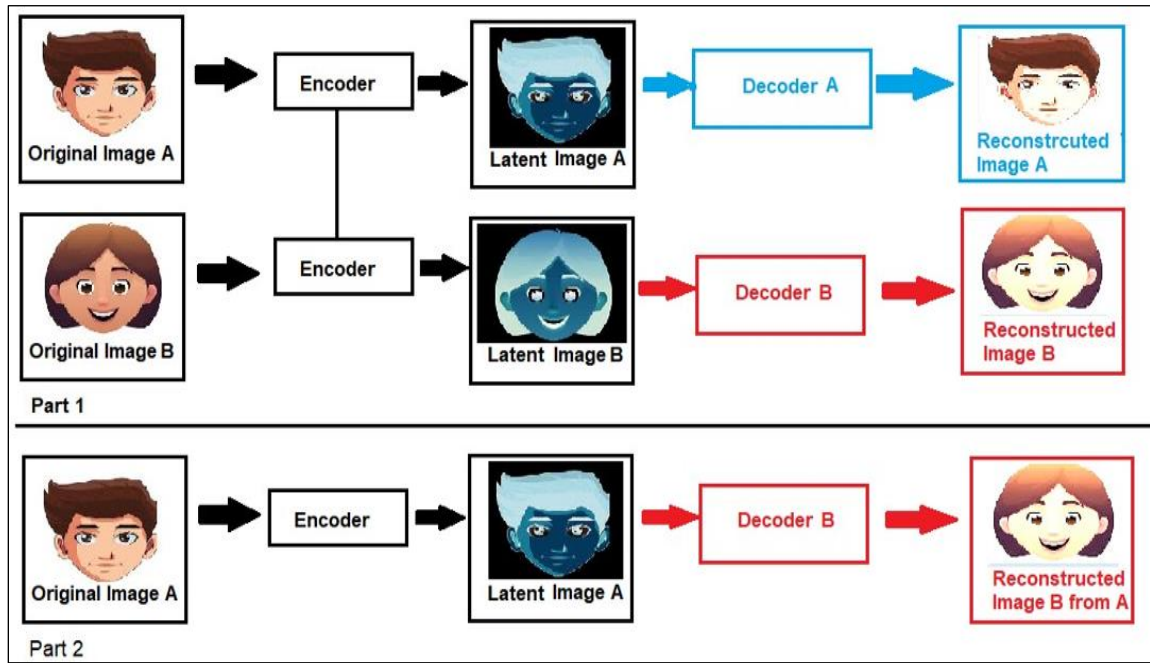


Figure 2.5: Deepfake creation process using autoencoders and decoders

Sabir et al. [167] have leveraged CNN models with several variations such as DenseNet, DenseNet+Alignment, DenseNet+Alignment+BiDir, ResNet50+Alignment, ResNet50+Alignment+BiDir, and ResNet50 on four face manipulation benchmark datasets FaceForensics++, Deepfakes, FaceSwap, and Face2Face. In addition, they used two face pre-processing techniques, i.e., landmark-based alignment and spatial transformer network, to detect Deepfake video streams and found the DenseNet+Alignment+BiDir method efficient for detecting face manipulation in videos.

Li et al. [165] developed a Long-term Recurrent Convolution Network (LCRN) system based on CNN and LSTM to detect Deepfake videos. Their method was based on eye-blinking behavior. In forged videos, faces with open eyes are shown, i.e., the eye blinking rate is less than normal human eye blinking. The authors used CNN model to extract facial and eye area features and LSTM to learn temporal sequential patterns.

Jung et al. [184] developed the DeepVision algorithm, which functions in adjacent frames

to detect and track eyes. The algorithm is used to compute eye measurement through the target detector in the pre-processing of input data, and the eye tracker monitors the blinking of the eyes. Later, the calculated data was correlated with the database of Deep Vision. The blinking pattern varies with sex, age, and cognitive activity and goes with the daytime also. The proposed DeepVision algorithm has tracked specific improvements employing several ML algorithms and a heuristic approach to check Deepfakes credibility with an accuracy of 87.5%. The algorithm's drawback is that blinking is associated with mental illness and dopamine activity and couldn't apply to people with mental illness as most Deepfake detection approaches use individual video frames and cannot learn from temporal data.

Lima et al. [180] proposed a detection method incorporating temporal information and several popular networks that took advantage of temporal features like RCN, R2PLUS, I3D, MC3, and R3D. They trained the extracted features on the Celeb-DF dataset starting from the pre-trained published weights. Each method was trained for over 25 epochs, and the best ROC- AUC score and test accuracies were recorded. The performance of video-based methods against a selection of techniques that only worked on frames and were not learned from temporal information was compared with those trained using temporal information. The classical frame-based method achieved an accuracy of only 66.8% with DFT in testing. The temporal information helped these network methods beat the latest baseline frame-based methods; in particular, R3D outperformed the other networks with a ROC-AUC of 99.73% and an accuracy of 98.26%.

Qi et al. [185] presented a DeepRhythm technique for Deepfake detection by noticing heartbeat rhythms, i.e., periodic skin color changes due to blood pumping through the face. In fake face videos, the usual heartbeat rhythms are disrupted or broken and are considered a strong predictor measure for Deepfake detection. The approach was further tested on FaceForensics++ and DFDC-preview datasets and was effective over other profound Deepfake detection techniques. Later, Hernandez-Ortega et al. [181] considered remote Photoplethysmography (rPPG) to measure heart rate to detect Deepfake videos. Their DeepfakeON-Phys approach used a Convolutional Attention Network (CAN) to retrieve spatial and temporal information to detect Deepfake videos. Their approach comprised a motion model to detect changes between consecutive frames and appearance models to

analyze each video frame's static information. Later, the motion model compared two adjacent frames from the appearance model's output data to provide the final output. The approach evaluated on Celeb-DF v2 and DFDC preview datasets and gave better results on the Celeb-DF v2 dataset with an accuracy of 98.7% and AUC of 99.9%.

Neekhara et al. [212] focused on the vulnerabilities of Deepfake detection methods from a practical perspective and emphasized the need for more robust detection methods. In a black-box environment, adversarial models that did not fully know classification models were used to target Deepfake detectors. The authors designed a transferable black-box attack that introduced a practical threat to the security of Deepfake detection. They performed experiments on the Deepfake Detection Challenge (DFDC) dataset to demonstrate that they could be easily evaded in the hands of a practical attack scenario by designing transferable and existing adversarial attacks.

Jiang et al. [213] compared their end-to-end face-swapping framework with other baselines over the deeperforensics-1.0 dataset comprising 60,000 videos with 17.6 million frames for face forgery detection.

Gandhi & Jain [214] emphasized adversarial perturbations to deceive Deepfake detectors using Face Gradient Sign and the Carlini and Wagner method by decreasing their detection accuracy. Further, they studied Lipschitz regularization for increasing robustness to input perturbations and Deep Image Prior (DIP) to improve Deepfake detectors. The DIP method eliminates perturbations using Generative Convolutional Neural Networks (GNNs) in an unsupervised method. Regularization enhanced the detection of perturbed Deepfakes. The DIP method achieved 95% accuracy on perturbed Deepfakes that misled the original detector, holding 98% accuracy in other cases.

Montserrat et al. [182] discussed automatic weight face regions and boosting techniques to detect face manipulation within videos by combining CNN and RNN. Transfer learning is performed using the DFDC dataset and ImageNet. Their approach works in three steps; first, MTCNN is used for face detection, then feature extraction is done by extracting visual and temporal features from the face using CNN. Finally, automatic face weighting layers (AFW) and GRU were used for prediction estimation.

Tursman et al. [215] proposed a social verification framework comprising PCA and hierarchical clustering techniques to analyze facial geometry consistency across a set of videos. They created a multi-view dataset of 25 human speakers and Deepfakes videos for each original video by manipulating mouth as speech changes using LipGAN. Later, using the simple mouth, Discrete Wavelet Transformation (DWT), and their PCA+hierarchical clustering system, they detected Deepfakes with an accuracy of 75% with their approach. However, due to the distinctive mouth signal, their approach is not robust when the mouth size is relatively limited compared to video size.

Hasan & Salah [118] have employed a blockchain-based Ethereum smart contracts framework to deal with media content authenticity. Their system preserved all historical information related to the creator and publisher of the digital media. The framework used an InterPlanetary File System (IPFS) to acquire media content and metadata in a decentralized manner. The system checks the authenticity of video content by tracking whether it is from some reliable or trustworthy source or not. The summary of various AI-enabled multimedia (Deepfakes) spam detection techniques is mentioned in Table 2.6.

Table 2.6: Summary of AI-enabled multimedia spam detection approaches

Reference	Dataset used	Features used	ML/DL approach used	Techniques/methodology used	Merits	Demerits
[164]	Image dataset (810 images and 928 spam images)	Image features	DL	CNN	Automatic image feature generation	The framework can be tested using RNN and LSTM
[96]	Deepfake videos; VidTIMIT database	Audio-visual features	DL, Face recognition methods, SL, USL	GANs, PCA, LDA, IQM, SVM, VGG, Facenet	Created Deepfake dataset from VidTIMIT database	Applied existing face recognition approaches for Deepfake detection.
[166]	300 Deepfake videos from video-hosting websites; HOHA video dataset	Image frame-level features	DL	CNN, RNN, and LSTM	Provided accurate results even on small-length videos	Worked only on intra-frame and temporal variations
[167]	FaceForensics++ dataset [101]	Semantic, temporal, Spatio-temporal features.	DL, Face alignment approaches	CNN, RNN, ResNet, DenseNet +Alignment+ BiDir	Their technique provided improved accuracy	Evaluated results on medium-quality videos only
[165]	(CEW+EBV) Eye blinking dataset	Eye blinking features	Physiological Signals, DL	LCRN (CNN+LTSM), EAR, VGG16	Used rate of eye blinking to detect Deepfake.	The model can be improved in case of frequent eye blinking or for

						videos where eyes are closed.
[183]	DF-TIMIT [96], UADFV [216]	Face-warping feature	DL	CNN (VGG16, ResNet50, ResNet101, ResNet152)	Robust and efficient	Physiological signal may not be notable in frontal faces
[118]	Digital content	Unique hash address generated by IPFS storage, reputation system, Ethereum Name service, artist's profile.	Blockchain	Ethereum smart contracts	Verify the authenticity of the video by tracking its creator and publisher information.	The system can be enhanced to check fake content such as audio, images, text documents, etc.
[211]	VISOB, Celeb-DF dataset [5]	Biometric comparison of the eyebrow region to detect face-swap	DL	LightCNN, ResNet-50, DenseNet-121, SqueezeNet	Best results on the Celeb-DF dataset	Require in advance biometric enrollment of people
[180]	Face-Forensics++ [101], DFDC [217], DF-TIMIT [96], UADFV [216], Celeb-DF V2 [218]	Spatio-temporal features	DL	XceptionNet architecture with metric learning; RCN, R2Plus1D, I3D, MC3, R3D methods	Used temporal information to make network decisions.	Limited DL-based image classification methods outperformed temporal features.
[185]	FF++ (FaceSwap) FF++ (Deepfake) DFDC Preview [217]	Physiological features	DL	CNN + Attention Mechanism	Employed dual-spatial-temporal features to vary face	It can be extended with other existing Deepfake detectors.

					and fake types dynamically.	
[181]	Celeb-DF V2 [218], DFDC Preview [217]	Physiological features- heart rate signal	DL	CNN + Attention Mechanism	The method used is unique and has a lot of potential	The robustness of the approach can be analyzed against face manipulation techniques and temporal data incorporation.
[184]	Deepfake dataset	Physiological features (Eye blinking)	DL	DeepVision algorithm	Physiological features solved the drawbacks of integrity verification algorithms.	It cannot be used for people with mental illness.
[214]	Dataset (5000 real and 5000 fake images)		DL	Lipschitz regularization, DIP, VGG-16, ResNet-18	DIP method shows good results	DIP defense takes a lot of time to process an image. Robustness for Deepfake detectors can be further improved to adversarial perturbations.
[215]	Multi-view video dataset	Video frames (mouth landmarks)	USL, DL	Hierarchical clustering - (Simple mouth, DWT), CNN, LipGAN	Robust in case of different camera angles.	The robustness of the model depends on the mouth size. Work can be extended with

StyleGAN.

[182]	DFDC dataset [217]	Visual and temporal features	DL	Conv-LSTM; Conv-LSTM + MTCNN; EfficientNet-b5; Xception; GRU	Made robust predictions using weight face regions automatically and boosting techniques.	Ignored audio content, which could further improve detection accuracy.
[212]	DFDC dataset [217]	Video frames	DL; attack models	CNN; universal attack, white-box attack, black-box transferable attack	Found that BlackBox attacks and universal attacks are a threat to Deepfake detectors.	Emphasized the need for a robust Deepfake detector

2.3 Spammer Account Detection Approaches

The OSN users are engaged by various spammer accounts to disseminate spam and other cybercrimes. Thus, several spammer account detection approaches are categorized into spammer accounts, spambots, and spammer community detection techniques are reviewed in the following subsection:

2.3.1 Fake/Compromised and Cloned/Sybil Account Detection

Zhu et al. [219] implemented a scalable supervised matrix factorization technique using Social Regularization (SMFSR) for spammer discovery by analyzing users' social relations and actions. The SMFSR technique was tested on the NASDAQ: RENN dataset from Renren.com and compared with the state-of-the-art matrix factorization methods for performance analysis.

Zheng et al. [133] provided a supervised ML-based solution on a Sina-Weibo dataset comprising 30116 users and 16 million messages to group user account into spammers and non-spammers. They constructed a labeled dataset and extracted features from text messages and users' social behavior. They used the SVM algorithm on the extracted features from the labeled dataset set to provide better results than other approaches, such as NB, DT, and Bayesian Network (BN).

Xiao et al. [178] provided a scalable supervised ML framework to detect fake account groups or legitimate by the identical actor in OSNs. A single malicious user may have created thousands of bogus accounts to spread spam and exploit the system for maximum genuine users reach. The proposed method was applied to a LinkedIn-grouped dataset that detected more than 250,000 fake accounts with Area under the curve (AUC) of 0.98 and 0.95 for the out-of-sample test set.

Ruan et al. [29] detected compromised accounts in OSNs by analyzing the social behavior of OSN users. For this, they analyzed user clickstreams on the OSN website to recommend social behavior features to characterize user social activities on OSNs and extracted eight behavioral features from extroversive and introversive activities performed by users. The fake profiles are identified because they cannot fit into a legitimate user behavior profile.

Their approach accurately detected compromised accounts when tested on sample Facebook users with complete and accurate behavior profiles.

Singh et al. [53] proposed a novel approach using ML classification techniques to classify traders dealing with sales and purchasing fake or compromised Twitter accounts (Spammers) from legitimate users. For analysis, they used NodeXL to gather 15,750 tweets concerning Twitter followers' business posted by 995 users based on their graph and content-based features. They also employed five different classifiers to classify users and spammers and inferred that the RF classifier outperforms with an accuracy of 85.2% to detect spam.

Zhang et al. [156] proposed a framework based on SSL for social spam detection by combining co-training with a k-medoids clustering algorithm. K-medoids clustering algorithms gather useful samples for labeling as early seeds. Later, they used the co-training classification of content and behavior features to detect social spammers. The framework result was better than many SL algorithms, such as NB, RF, JR, J48, Bagging, and LogitBoost.

Fu et al. [161] developed a framework to differentiate between spammers and authentic OSN users by combining supervised and unsupervised ML algorithms with temporal factors to calculate the deviation in users' activities and evolutionary patterns. They tested their framework on real-time datasets and found it efficient for classifying different spammers and legitimate users.

Cao et al. [27] implemented a forwarding message tree approach that recognized internal relations based on communication between the sparsely connected hidden suspicious accounts and forward messages by initiating the forwarding message tree concept to remove the identified accounts simultaneously. Further, they compared various classifiers trained from six specified features: average tree weight, propagation depth, repeated forwarding behavior, propagation speed, and propagation breadth, forwarding layer relation.

Shen et al. [160] proposed a generalized classification framework by considering Multi-

View Learning for Social Spam detection (MVSD) that integrated social information based on Negative Matrix Factorization (NMF) to consider different and variable strategies used by spammers that were not described by single views of information for detecting spammer accounts on the massive Twitter dataset.

Rathore et al. [220] proposed an effective SpamSpotter framework on Facebook that classified spammers from genuine users. A mechanism is developed to collect a dataset based on Facebook user profiles and content features. The proposed framework SpamSpotter has an Intelligent Decision Support System (IDSS) that leveraged ML classifiers such as DT J48, Jrip (JR), BN, RF, SVM, K-Nearest Neighbor (KNN), LR, and Decorate (DE) on the dataset. Their framework provided better accuracy, Matthews Correlation Coefficient (MCC), and F-Score than existing approaches.

Lorimer et al. [221] proposed a system to discover participatory identity theft in mobile OSNs by analyzing each follower's abnormal behavior without considering semantic analysis. The decision tree with weights and corresponding thresholds is rearranged automatically with an aggregator node to achieve an increased response and success ratio.

Singh et al. [168] detected five types of spammers or malicious users in Twitter microblogging OSN, i.e., sole spammers, follower market traders, fake, compromised profiles, and pornographic users. The spammer classification uses trust-based, user-based, and content-based features for ML techniques such as BN, LR, J48, RF, and AdaBoostM1. Further, trust calculation is done using the Klout Score (KS) over data of 20 million tweets of about one lakh Twitter users. Among all the mentioned classifiers, the RF algorithm detected spammers with an accuracy of 92.1%.

Barushka & Hajek [146] have used DNN learners with a bagging EL approach to filter spam and spammers on social networks. The authors used high dimensional data from the Hyves dataset's content and user behavior features to get better results than other techniques such as artificial immune systems, DT, NB, SVM, etc.

Jeong et al. [176] proposed a sorting scheme for Twitter SNSs to spot spammers using the social network properties in individual local OSNs. They hypothesized that spammers'

online relationships are dissimilar to legitimate users. Their hybrid approach (Cascaded-Filtering) used properties of both techniques, i.e., Triad Significance Profile (TSP) filtering and social status (SS) filtering in a two-hop subnetwork. The proposed approach was tested on the Twitter dataset and compared with the Collusionrank algorithm, which yielded 96.3% TPR and 5.7% FPR.

Sahoo & Gupta [140] proposed a Chrome browser extension system to detect bogus Twitter profiles by gathering Twitter data through the Twitter API and then analyzing each Twitter profile using a traceback method to determine the source of fake content to examine different features to recognize fake user accounts in the ML environment. Later 2020, they proposed a hybrid approach to differentiate between spammer and non-spammer accounts. Their system used a GA optimization algorithm for analyzing and choosing the best features to train ML classifiers such as SVM, neuro-fuzzy classifier, RF, and found RF with GA performed better with a detection rate of 99.6% [139].

Al-Qurishi et al. [179] have reviewed several Sybil detection techniques categorized into graph-based, ML-based, prevention-based, and manual verification-based. Later, they proposed a systematic model with three components, i.e., data gathering, feature extraction, and a deep-regression model, to predict Sybil attacks with 86% accuracy [179].

Fu [222] has made a Sybil detection system with a weighted linear model and a voting-based classifier. Their system integrated the results of several existing Sybil detection methods, such as SybilInfer, SybilDefender, SybilRank, and SybilExposer, etc., to improve prediction accuracy.

Gao et al. [223] implemented the SYBILFUSE framework for Sybil detection using local classifiers to compute trust among nodes and edges, later disseminating classification scores through a weighted score propagation using the global structure. The framework was tested on the Twitter network with 20M nodes and 265M edges and attained 98% Sybil coverage among top-rank nodes. The summarized information related to Fake/compromised and Sybil/cloned account detection techniques are mentioned in Table 2.7.

Table 2.7: Summary of Fake and Sybil/cloned account detection approaches

Refer ence	Dataset used	Features used	ML/DL approach used	Techniques/ Methodology Used	Merits	Demerits
[219]	NASDAQ: RENN	User-based and behavior-based SL features	SL	MF, SVM, SMFSR	Scalable approach	Other important hidden factors can be explored using tensor factorization techniques.
[133]	Sina-Weibo dataset using Sina-Weibo API based	Message content-based, user-	SL	SVM, NB, DT, BN	SVM performed better than NB, DT, BN	Used manual feature extraction
[178]	LinkedIn dataset	Distribution features, pattern features, frequency features	SL, EL	Scalable SL pipeline framework using SVM, RF, and LR	Scalable approach	Limited features and clustering algorithms used
[29]	Built dataset of user activities using Facebook.com	Social behavior features (Extroversive and Introversive)	SD	Behavior variance	Provides better results for complete and accurate social profiles	It depends highly on social behavior patterns only
[176]	Twitter follow link dataset; follow spammer dataset	Triad (Isomorphic subgraph), social status (positive link probability)	Hybrid, EL	Cascaded filtering hybrid approach; Collusionrank algorithm	Secure and real-time	The framework can be tested with USL techniques
[156]	UDI- Twitter dataset	Content-based and behavior	SL, USL,	Co-training + k-medoids	Combined co-training	Worked on limited

		features	SSL, EL	clustering algorithm; NB, RF, JR, J48, Bagging, LogitBoost	with K-medoids for better results	features; Framework can be extended using AL and transfer learning
[53]	Twitter dataset using NodeXL open-source tool	Content-based (Retweet Frequency, posting Date of Retweets) and graph-based (Followers/ Following ratio, Reputation)	SL, EL	BN, LR, BC, RF, and AdaBoostM1	Analyzed Twitter followers spamming merchants	It used limited features only.
[27]	Sina- Weibo dataset using Sina-Weibo API	Forwarding layer relation, repeated forwarding behavior, average tree weight, propagation depth, propagation breadth, propagation speed.	SL, EL	Forwarding message tree approach; J48, RF, Random Trees, BN	Better feature analysis	Not real-time; It can be combined with graph-based algorithms
[160]	Honeypot dataset [224]; Kwak's dataset [225]	Text-based, URL-based, hashtag-based features	USL	MVSD framework with NMF approach	Better performance with small training data	The framework can be tested on a larger dataset
[220]	Facebook dataset using Facebook API	Profile-based, content-based	SL, EL	IDSS using ML classifiers: BN, RF, DE, J48, JR, KNN, LR SVM	It can be integrated with Facebook	IDSS can be extended with DL techniques
[168]	Twitter and Facebook dataset using Twitter APIs and Facebook	Trust-based, user-based, and content-based	SL, EL	BN, LR, BC, RF, J48, and AdaBoostM1	Detected (Sole spammers, pornographic users, follower market traders,	Automatic feature extraction techniques can be incorporated.

	APIs				fake and compromised profiles)	
[161]	Sina-Weibo dataset using Sina-Weibo API	Graph-based: Degree Centrality, Bidirectional link ratio, betweenness centrality, and local clustering coefficient. Non-Graph-based: # of tweets, # of tweets with hashtag, # of tweets with URL, and # of retweets/tweets with a reply	SL, USL	SVM –common, TrustRank, SVM- combined, and SVM-clustering, SVM combined with temporal factors.	The evolutionary temporal pattern of users was shown for spammer detection.	The offline detection system can be extended to form an online detection system. It relies mostly on labeled data; unsupervised techniques can be integrated.
[146]	Hyves dataset	Content-based features and user behavior features	SL, DL, EL	NB, k-NN, C4.5, MLP SVM, AdaBoost M1, RF, CNN, DNN, DNN+AdaBoost, DNN+Bagging, DNN with random subspace	The Model can detect both spammers and spam messages	Worked on a small dataset
[179]	Twitter dataset using Twitter API	Profile, emotion, graph, content, quality, temporal, and topic-based features	DL	DNN model using Google tensor-flow library	Provides better results on unclean and noisy data	Pre-processing and other DL-based techniques can be incorporated
[139]	Twitter profiles data	User account features	SL, MOA, EL	NB, LR, JRIP, Bagging, RF, PART, J48, GA	Most optimized features selected using the GA algorithm	The framework can be tested on other OSNs also.

2.3.2 Spambot Account Detection

Social bots are hard to detect, as they always try to make social relationships with legitimate users to avoid being discovered [40]. Existing ML/DL-based spam detectors for detecting spammer accounts are not proficient enough to identify spambots. Some of the recent developments for spambot detection are mentioned below:

Davis et al. [169] developed BotOrNot technique for providing a free service to check the bot-like characteristics of any Twitter account. Since 2014, nearly one million requests, single or multiple accounts, have been served by this technique. The BotOrNot classifiers work on the dataset extracted using the Twitter API and the existing Caverlee dataset [224].

Varol et al. [226] implemented a framework to detect different types of bots, such as spammers and self-promoters, on the Twitter platform by extracting huge features from Twitter's open data and metadata of users. They have also considered social connectivity and information flow to analyze user interactions. From the total number of active Twitter accounts, 9 to 15 percent are bot accounts. Classifiers have been trained on existing datasets and on the merged dataset to get a highly accurate result. They also performed a cluster analysis to find various types of bots on account of user behaviors.

Cresci et al. [175] mentioned social spambots with human-like characteristics that helped them to be unseen and undetected. They used digital DNA techniques to deal with OSN users' behavior by converting their behavioral lifetime into a sequence of characters. Using the Social fingerprinting technique, they defined similarity measures for these digital DNA sequences to categorize social spambots and genuine accounts.

Kudugunta & Ferrara [103] detected bot accounts on Twitter using a DNN-based LSTM framework. The framework received input as extracted contextual features from tweet content and metadata, providing 96% accuracy in classifying bots from humans.

Jr et al. [40] have implemented a wavelet model on users' textual patterns to retrieve spectral patterns to distinguish between social accounts, legitimate bots, and malicious bot accounts. Discrete Wavelet Transform (DWT) and Lexicon coefficient attenuation weighing scheme retrieve the writing style pattern from user posts content to create feature

vectors. The RF classification algorithm is applied over two datasets to classify human accounts, legitimate bots, and malicious bots on Twitter.

Walt & Eloff [174] discovered engineered features created by bots different from human-generated bogus accounts' characteristics. ML algorithms such as RF, SVM, and boosting models deployed on these features predicted a 49.75% F1-score among bot-generated and human-generated fake accounts.

Perna & Tagarelli [136] analyzed and ranked Twitter accounts based on their bot behavior using a Learning to Rank Social Bot framework (LTRSB). Their framework used feature selection methods such as PCA, IG, Gain Ratio (GR), OneR, CFS, and DT-J48 as learner-based feature selection methods to extract user, network, content, aggregate, and temporal-based features. The framework functioned with queries and ground truth information to identify several bot groups to provide better results than existing learning-to-rank methods.

Beskow & Carley [171] developed a Tier-1 method named Bot-hunter for bot detection and an event-based method for bot data annotation. Due to the availability of training data, the Tiered method is used in large Twitter streams to measure bot penetration. They built only the Tier-1 model and compared it with the existing botometer algorithm. Later, the same authors [132] employed a multi-tiered ML toolbox in their past Tier-1 research, 'Bot-hunter', for bot detection. They used content and network features extracted using a snowball sampling technique for better accuracy from a dataset of 50,000+ entities. They used supervised ML models to provide better prediction over tier-1 models with newly added features. The network features helped the model better detect, control bot behavior and make it difficult for bots to manipulate and escape.

Lingam et al. [172] developed a Deep Q-Learning (DQL) model for spotting social bots on Twitter using social attributes comprising tweet posts, users' profiles, and graph-based characteristics. The DQL model considered each social attribute of the user as a 'state' and the movements of the learning agent during state transition as an 'action.' The Q-value function found all state-action pairs to create the probability values for state moves between the various state-action pairs. A learning agent can detect a social bot by choosing the finest Q-value in each state. The most influential users are also identified using the top-k

influential algorithm based on tweets shared and users' interactions. The performance of the DQL model was 5-9% better in terms of precision when evaluated and compared with algorithms such as Regularized Deep Neural Network (RDNN), Deterministic Q-Learning (QL), and Feed-Forward Neural Network (FDNN) on three different datasets, namely User Popularity Band dataset, Fake Project dataset, and Social Honeypot dataset.

Mazza et al. [84] investigated malicious social bots by analyzing Twitter users' retweeting/re-sharing behavior. They implemented two techniques: a ReTweetTweet (RTT) visualization and the Retweet-Buster (RTbust). The RTT technique provided valuable insights from retweeting behaviors of Twitter accounts. The RTbust technique utilized an LSTM auto-encoder to convert retweet data into compressed latent feature vectors using a Hierarchical Density-based Clustering Algorithm (HDBSCAN). The clustering algorithm helped decide whether the online account was legitimate or a bot account. Moreover, they implemented the RTbust technique with other feature extraction techniques such as PCA, VAEs, and TICA. They found the RTbust technique best with the neural network-based feature extraction technique, i.e., VAEs based on precision, accuracy, F1 measure, and MCC.

Qi et al. [43] developed ML-based framework using an LR algorithm on content-based features to detect bot behavior on Twitter. Their dataset included a list of four bot accounts, i.e. (content polluters, fake followers, and social spambots) and a list of genuine user accounts. They analyzed and classified differences between bot behavior and human user account features.

Wu et al. [151] proposed upgrading the conditional generative adversarial network for imbalanced data sets before the final social bot classification. In their approach, the Gaussian Kernel Density Peak Clustering Algorithm (GKDPCA) is used to tackle noise during data gathering and data imbalance problems and Wasserstein distance to address model failure and gradient loss problems found in the older CGAN.

Wu et al. [152] proposed a novel framework DABot, i.e., based on a newly built RGA DNN model to detect social bots. The RGA model consists of a residual network (ResNet), Bidirectional Gated Recurrent Unit (BiGRU), and an attention mechanism. In addition,

they employed Active Learning (AL) to increase the labeled data efficiently. The summary of studies related to spambot account detection techniques is mentioned in Table 2.8.

Table 2.8: Summary of spambot account detection approaches

Reference	Dataset used	Features used	ML/DL approach used	Techniques/ Methodology Used	Merits	Demerits
[169]	Caverlee-2011 [224]	Sentiment-based, user-based, friends features, temporal-based, content-based, network-based	SL, EL	RF	Provided free service to check bot-like characteristics	It can be extended to work as a browser plugin
[170]	Varol-2017 [170]	User-metadata-based, friends-based, network-based, temporal features, content-based and sentiment-based	USL, EL	RF and K-Means	Extracted 1150 features and classified various types of bots	The framework can be tested with classification and clustering algorithms
[175]	Cresci-2017 [37]	Behavioral features	SL, USL	Social fingerprinting: Digital DNA	Considered collective behavior of a group of accounts rather than single behaviors.	Considered only the behavior features and not work on fake bot accounts.
[40]	Twitter single theme dataset and miscellaneous theme dataset	Corpus size, lexicon size, wavelet component (WC), wavelet magnitude (WM), wavelet domain score (WDS)	SL, EL	DWT, RF	Classified accounts based on their writing style.	Results can be improved using other ML Classifiers.

[174]	Twitter dataset using Twitter4J API	Identity, relationships, and behavior/ messages of the account.	SL, EL	RF, Adaboost, SVM-linear	Features generated from fake bot accounts are not similar to counterfeit accounts created by humans.	ML models applied to limited features.
[136]	Bot dataset (11k bot accounts, 7k non-bot accounts)	User, network, temporal, content, and aggregate-based features	SL, USL	LTRSB framework	Analyzed and ranked Twitter accounts based on their bot behavior	Worked on just two classes, i.e., bot or not bot.
[171]	Twitter dataset using Twitter Streaming API	Tier 1 features: user-based, network-based, content, and timing-based	SL	NB, LR, SVM, DT, and RF models	Flexible system, i.e., researchers can choose the model as per their requirements.	Provided only Tier-1 model
[132]	Twitter dataset using Twitter Rest and Streaming API	Network and content features	SL, SM	Multi-tiered ML-toolbox	New models developed are highly accurate and are difficult for bots to manipulate and evade.	Models will work only for low-volume applications. A multi-class model can be built to work on content, profile, and temporal features.
[103]	Cresci-2017	Features extracted from tweet content and metadata	DL	DNN-based LSTM framework	Metadata features are used to provide better results.	It worked on limited human and bot user-labeled examples.
[172]	User popularity band [227], Fake project	Social attributes (tweet posts, user profile, and	DL	DQL model, FDNN, RDNN	Worked on features other than profile or behavioral	The model can be extended with better NN algorithms

	[37], and social Honeypot dataset	graph-based)				
[84]	Cresci-rtbust-2019 [84]	Latent feature vector extracted from retweeted data (RT users entropy, RT days entropy, RT rate, RT days, etc.)	SL, USL, DL	RTbust, HDBSCAN	Discovered suspicious retweeting patterns from the large dataset	Precision and recall can be improved by pruning and growing clusters by adding more borderline accounts
[152]	Sina Weibo labeled dataset	Metadata, interaction- based, content-based, timing-based	SL, DL, AL	DABot framework with RGA model (ResNet block, BiGRU block, attention layer, inference layer); SL- (LR, SVM, ELM, RF), DL- (MLP, LSTM, ComNN, CNN)	Used AL to label large user data at less cost- efficiently.	Work can be extended by detecting clusters of social bots.
[151]	Human accounts and social bot account dataset [170], [175]	user meta-data, sentiment, friends, content, network, and timing	DL, USL	CGAN, GKDPCA, Wasserstein distance	Worked on data imbalanced and noise problem	Work can be extended with other OSNs with the network, behavioral and sequence features.

2.3.3 Spammer Community Detection

Spammers form large communities of linked accounts that are used to distribute spam. Therefore, detecting those spamming communities that reside in OSNs is extremely valuable. Bindu et al. [173] developed a SpamCom system to discover spam communities connected to spam accounts to blow out spam to a broad set of genuine users. Their method identified overlapping community-based features as hypergraphs to detect spammers with content similarity, age, community-based, graph, and URL features from user profiles. Later, they recognized spammers are grouped to identify hidden spammer communities in Twitter OSN.

Fazil & Abulaish [177] built a hybrid approach to spot automatic spammers by merging community features with metadata, content, and interaction-based features. They used nineteen spammer features based on their neighbor nodes and their interaction network as malicious users, which are rarely followed by legitimate users but randomly followed by spammers. Also, they used three classifiers, RF, DT, and BN, and detected low edge density among the various followers and followings.

Gupta et al. [163] discussed spammers that use phone numbers on Twitter to blow out spam campaigns. In the proposed approach, Twitter is used as a heterogeneous network with heterogeneous nodes and their connections to detect spammers using a classification approach with a feedback-based AL strategy with the Hierarchical Meta-Path Score (HPMS) metric to calculate the closeness of unfamiliar users to well-known spammers. Compared with other baseline methods that use two-class classifiers, such as LR, Latent Dirichlet Allocation (LDA), KNN, DT, NB, RF, and SVM, the proposed approach performed better during small training instances to detect spam in terms of F1-score and AUC.

Cruickshank & Carley [159] suggested a Multi-View Modularity Clustering (MVMC) technique for investigating hashtags and Twitter discussions during the COVID-19 pandemic. The authors found significant shifts in clusters for various temporal and topical trends during COVID-19 Twitter discussions. The MVMC technique delivered good results with a scalable and incomplete view of data but can be further improved with cleaned and real-world large data.

Hao & Zhang [141] built a DL-based ensemble approach DCEDM to classify shilling attacks by measuring user-user graph edge weight based on user behavior similarity score. The feature extraction is performed using SDAEs to produce several clustering outcomes. Besides, they used Persistence Optimization Algorithm (POA) for community detection. Their approach performed better than other supervised and unsupervised baseline detectors over Netflix and Amazon datasets in various attack scenarios. The summary of spammer community detection techniques is mentioned in Table 2.9.

Table 2.9: Summary of spammer community detection approaches

Reference	Dataset used	Features used	ML/DL approach used	Techniques/ Methodology Used	Merits	Demerits
[163]	Twitter dataset using Twitter streaming API	Phone Numbers + OSN base features (URL, hashtags, hub scores, etc.)	SL, EL	HPMS + Active learning method compared with ML classifiers: LR, LDA, KNN, DT, NB, RF, and SVM	Used HPMS and AL with a feedback mechanism	Can be tested on other OSNs with more features
[177]	Twitter dataset	Metadata, Content, Community, and interaction-based	SL, EL	RF, DT, and BN	Classified spammers based on their interaction network.	Temporal features can be considered.
[173]	Honeypot dataset [224]	Community-based features, graphs, user account features, user behavior, topology-based features, and content similarity	SL, USL	Unsupervised SpamCom approach; ADTree, J48, KNN, SVM, and NB	Identified hidden spam communities	Work can be extended with real-time streaming data and user profile features.

		features				
[159]	Twitter dataset	Co-occurrence ofUSL hashtags text and URLs		MVMC	Performed well with scalable and incomplete data.	MVMC technique can be improved with cleaned and real-world data.
[141]	Netflix and Amazon review dataset	Graph features	SL, USL, DL, MOA	POA, DCEDM	DCEDM outperformed other baseline detectors.	Proposed algorithms can be further optimized.

2.4 Summary

Cyber-attacks have become more complex each day, and social engineers are deploying novel and ingenious deceptive approaches. So, there is a need to counter social engineering attacks with robust detection systems. This chapter reviewed traditional and recent ML and DL-based techniques/approaches for spotting social spam and spammer account detection techniques/approaches. The shortcomings of conventional approaches, such as dataset collection, feature inconsistency, accuracy, faster performance, and so on, have been resolved using ML and DL techniques.

The data acquired from OSN using APIs and web crawlers can be biased as they cover some specific category of language, content, and account information. Moreover, streaming data from OSN is incomplete, redundant, and involves missing values, etc. Manual preprocessing is not feasible. Thus, several dimensionality reduction approaches are used to provide ML/DL models with prominent features. Several feature analyses used to detect spam in OSNs are categorized as content-based, user/profile/behavioral analysis, social network analysis, topological/network/graph-based, statistical, and temporal and physiological analysis.

For text-based spam detection, SL techniques have been used extensively. However, SL techniques are ineffective due to their intrinsic challenges, such as lack of updated labeled training data, manual labeling, class imbalance problem, spam drift, etc. USL techniques were used to identify clusters of spammers, bots, or spam campaigns. However, they are not productive with large OSNs as they require large data storage. SSL techniques provided a better result in the absence of publicly labeled data and have been used to circumvent the limitations of SL and USL. SL classifiers are combined with USL models to train the labeled and unlabeled data.

The DL models are recommended over ML models due to better feature extraction and faster spam detection over OSNs. However, DL models also have some shortcomings. DL models usually require more data than standard ML models. For better training and testing of ML/DL models, large-scale ground truth data is needed. It's observed that most of the ML/DL-based frameworks for social spam detection were trained and

evaluated on a small dataset and imbalanced datasets. Therefore, collecting and analyzing a big, unbiased dataset for real-time social spam identification remains a significant challenge.

Moreover, the existing studies also revealed that existing DL-based spam detection approaches have not used recent word representation/pre-trained word embeddings to generate features. Some studies have used traditional word embedding techniques such as BoW, TFIDF, Word2Vec, etc. for feature generation. The traditional word representation techniques are time-consuming and inefficient in generating word vectors. Thus, requirement is to incorporate recent pre-trained word embeddings in the DL-based approaches.

Although ML and DL-based spam detectors have seen exceptional success and high accuracy, they are still vulnerable to multiple adversary attacks. Most of the suggested approaches do not address adversaries attacking the defense mechanism. Thus, the robustness of Deepfake detectors can be further improved for adversarial perturbations. Further interesting areas to investigate include hybrid approaches, data sampling techniques, advanced word embeddings, performance issues, and security against coordinated inauthentic behavior and adversarial attacks.

Chapter 3 : Proposed PHVE and PSVE Models using Machine Learning

3.1 Introduction

Cybercriminals exploit social networks to spread spam, misleading information, fake news, and malicious links. Misinformation in the form of posts, fake news, rumors, etc., about medications, immunizations, surgical masks, etc., has recently appeared on social media regarding the COVID-19 pandemic, endangering individuals' mental health and posing additional challenges for the government if not detected promptly [159], [228]. Hence, it is crucial to detect such deceptive social media spam.

Traditional methods for spam detection, such as URL list-based filtering and honeypot/net mechanisms, are ineffective. ML-based techniques overcome the limitations of traditional spam detection methods and have been extensively researched in the last decade. A recent review of social spam detection techniques revealed that most researchers have only evaluated spam detection techniques on small and balanced datasets [229]. The existing spam detection approaches cannot perform better on the imbalanced dataset, i.e., one class has inadequate proportions compared to the other class. This issue, known as class imbalance, is considered one of the main issues in ML classification [230]. Most ML algorithms assume that the distribution of data samples across class labels is roughly proportional, making learning difficult since they are biased towards the majority group and predict the majority group's class in most cases. The class imbalance problem can be solved using data resampling techniques. Besides, the performance of ML models is challenged due to noisy data, irrelevant features, spam drift, etc.

The major findings of this chapter have been peer-reviewed and published in:

- Rao, S., Verma, A. K., & Bhatia, T. (2023). Hybrid Ensemble Framework with Self-Attention Mechanism for Social Spam Detection on Imbalanced Data. *Expert Systems with Applications*, Elsevier, 217, 119594. [I.F. 8.665]. SCIE.

This chapter presents a framework for social spam detection using ensemble ML models. The performance of social spam detection is improved by leveraging improved data preprocessing, data balancing, and voting-based ensemble techniques. In the proposed framework, the imbalanced datasets are balanced using NearMiss and SmoteTomek techniques to feed several machine-learning models. Later, the baseline ML models and proposed voting-based ensemble models are evaluated on imbalanced and balanced datasets by adjusting hyperparameters. The performance of ML models is found to be improved on balanced datasets. Experiments and comparisons with existing ML-based approaches/techniques reveal that the proposed voting-based ensemble models produce significantly better performance.

3.2 Related work

Traditional approaches for spam detection, such as URL list-based techniques and honey-net-based systems, have several shortcomings, including data collection and portability, legitimate spammer IP addresses and small URLs, attribute discrepancy, adaptability, performance, and proficiency. Most of these shortcomings have been fixed by ML/DL-based approaches [229]. The following section briefly summarizes the recent research on textual social spam detection using various ML-based approaches.

For ML/DL-based social spam detection, the raw data is first collected using OSN APIs to form a dataset. The raw data is pre-processed to reduce noise and missing data. Feature extraction approaches are applied to retrieve only dominant features for ML/DL models. Common feature extraction techniques are TF-IDF, BoW, Word2Vec, etc. In BoW, n-gram text features are extracted from a group of words from the textual content.

Aiyar & Shetty [231] have detected spam comments on YouTube using character grams, word grams, and several ML algorithms such as Naïve Bayes (NB), random forest (RF), and support vector machine (SVM). Eventually, they found character grams are better than word grams to measure spamminess in YouTube comments and recommended using Word2Vec word embedding for better results.

Alberto et al. [232] have distinguished between spammers and legitimate users using

the SVM classifier on YouTube using social and anti-social behavior features. Gupta et al. [142] developed an ML framework with a BoW feature extraction technique to detect spam tweets. Mehmood et al. [158] have used TF-IDF text features to build a spam comment prediction model using ensemble learning.

Ahmed et al. [131] have used n-gram analysis with TF-IDF to detect opinion spam using Metaheuristic Optimization Algorithms (MOA) and ML models. In addition, they implemented six ML classifiers on the fake news dataset and found that accuracy improved with the number of features and declined with an increase in the size of the n-gram. Adding dimensionality reduction to ML models enhanced their performance. Most spam detection research relies on feature extraction methods such as n-gram, BoW, TF-IDF, and Word2Vec. Some commonly used supervised learning (SL) models for spam detection are NB, SVM, RF, and Decision Trees (DT).

Kardaş et al. [233] have tested the performance of several ML models, such as NB, SVM, LR, and neural networks, on the Twitter dataset using pre-processing and word representation techniques.

Saumya & Singh [157] have implemented an opinion mining framework using RF, gradient boost (GB), and SVM models to detect spam reviews on balanced datasets generated using Smote and Adaptive synthetic sampling (ADASYN) as data balancing techniques. Few other relevant studies have worked on the class imbalance issue and resolved the problem using under-sampling and over-sampling approaches. Some of the data resampling techniques used are random over-sampling (ROS), NearMiss, Synthetic Minority Over-sampling Technique (SMOTE) [234], ADASYN, Smote with boosting (SmoteWB) [230], [235]–[237]. Few studies have used semi-supervised learning (SSL) techniques over supervised and unsupervised to train unlabeled data [51], [144], [156], [162]. Some of the challenges identified after an in-depth review of related articles are as follows:

- Dearth of large datasets: ML/DL-based spam detection models were often trained on small datasets.
- Class imbalance problem: ML algorithms work best for the majority class while neglecting the minority class. Hence, balancing the imbalanced dataset using

under-sampling and over-sampling strategies is essential to avoid biased ML model training and testing.

- Spam drift: Spammers often change tactics to evade spam detectors. This degrades the performance of ML/DL-based spam detection systems.
- Performance: Building efficient and robust spam detector systems requires better data pre-processing, NLP-based pre-trained word embeddings, ensemble approaches, and advanced DL models.
- User privacy and security: It is necessary to build a safe and spam-proof system that examines text entering OSN users' profiles and stops it if it is spam to improve privacy and security.

3.3 Contributions

The following contributions have been made to determine the most efficient approaches for detecting spam for imbalanced data:

- Larger corpus is created by merging three benchmark datasets: SMS spam, Ling spam, and Twitter fake news.
- Data pre-processing and exploratory data analysis (EDA) are done on the larger corpus (*Combined dataset*).
- Dataset balancing techniques such as NearMiss and SmoteTomek are employed to balance the imbalanced dataset.
- Bag of words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) word representations convert text into numbers.
- Proposed two ML-based voting ensemble models (PHVE and PSVE) that leverage the predicted results of individual ML models for getting better performance.
- Performance comparison of proposed models (PHVE and PSVE) is done with various baseline models on imbalanced and balanced datasets using various performance measures.
- Finally, the performance of PHVE and PSVE is compared with existing approaches/frameworks.

3.4 Proposed Methodology

The overall proposed framework with ML and DL approaches for social spam detection to categorize the text message as *Spam* and *Ham* is depicted in Figure 3.1.

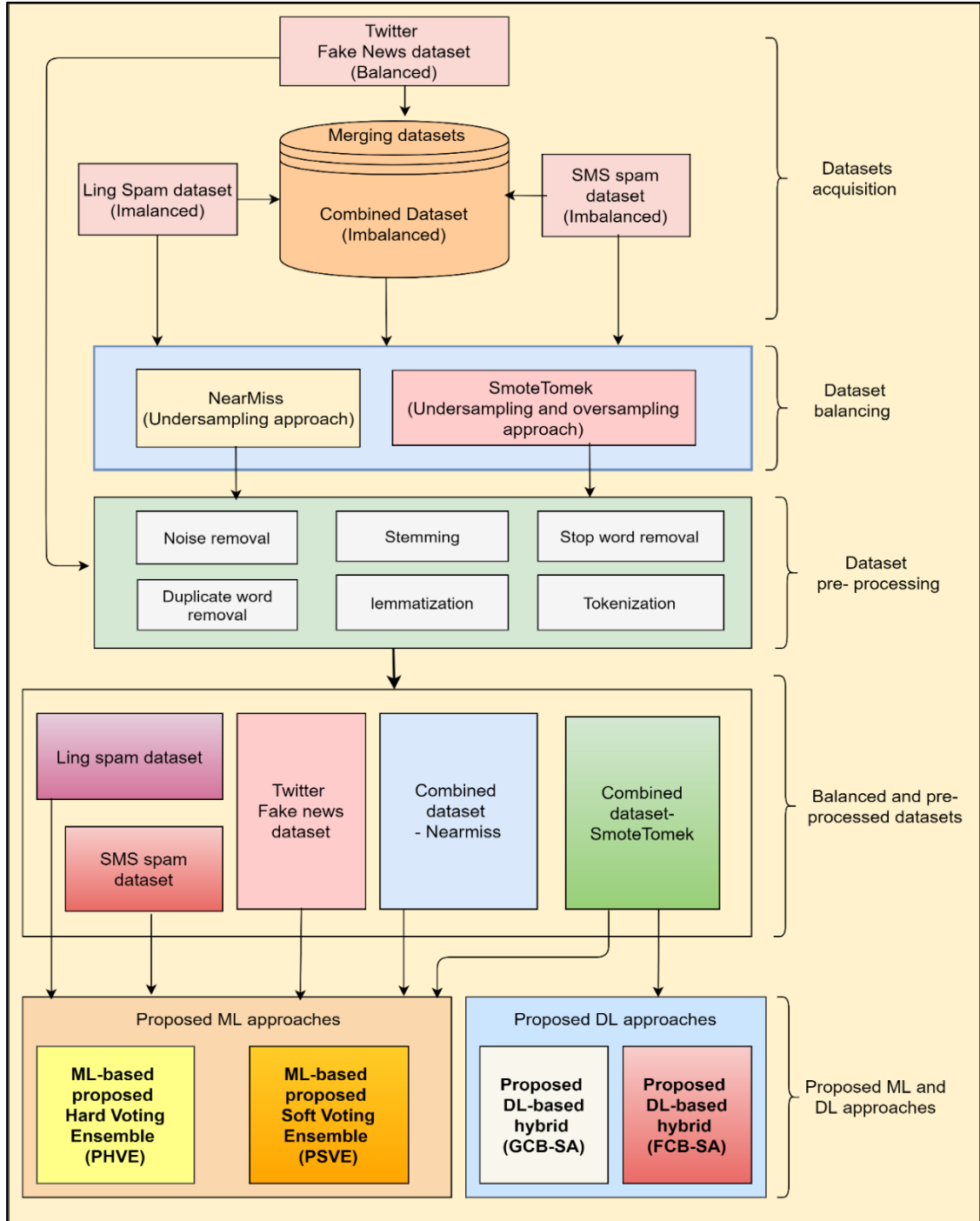


Figure 3.1: Proposed framework with ML and DL approaches for social spam detection

For ML-based approaches, two voting ensemble models (PHVE and PSVE) are proposed. The proposed methodology for building ML-based voting ensemble models consists of the following steps:

3.4.1 Dataset Acquisition

Due to the lack of a bigger dataset, we have built a larger text corpus of spam and ham text messages by merging the existing datasets from three different channels, i.e., mobile SMS, linguistics messages from email, and COVID-19 spam tweets from Twitter. Moreover, this will improve the training of DL-based models with a larger and more diverse dataset containing various forms of text spam on social networks. The three datasets merged are the *SMS spam*, *Ling spam*, and *Fake news dataset (FND)* to form a larger dataset named the *combined dataset*. The *combined dataset* comprises various spam content, i.e., text messages spam, research spam, job posting spam, healthcare-related spam, lottery spam, malicious links, free software download spam links, etc.

The merging phase involves challenges such as data heterogeneity, deduplication, eliminating irrelevant columns, and standardizing class labels. All of the mentioned challenges are handled by thorough data profiling, data cleaning, standardization, transformation, and deleting extra/irrelevant/empty columns such as the ‘ID’ column from the Fake news dataset, ‘Source’, and ‘Subject’ from the Ling spam dataset. Later, the content aggregation is done by appending data from three datasets under appropriate columns. Standardizing class labels is another important challenge where the class labels are given in different forms in the datasets. For example, in the SMS spam dataset, class labels are shown as ‘ham’/‘spam’; in the Ling spam dataset, the class labels are given as ‘0’/‘1’, and in the Fake news dataset, class labels are mentioned as ‘real’/‘fake’. All the class labels are converted to the same standardized form of ‘ham’/‘spam’. Later, the standardized labels are converted into (0/1) numeric values. The sample of the *combined dataset* is given in Table 3.1.

Table 3.1: Sample of *the combined dataset*

Text	Label	Label_text	Source
Free entry in 2 a weekly comp for a chance to win an ipod. Txt POD to 80182 to get entry (std txt rate) T&C's apply 08452810073 for details 18+	1	spam	SMS spam dataset
He telling not to tell any one. If so treat for me hi hi hi	0	ham	SMS spam dataset
hey there interested in some free xxx site 's ?	1	spam	Ling spam dataset
i do n't know about eskimo words for snow , but scottish gaelic has a special word " turadh " for when it stops raining ! compare : tha an t-uisge ann . is the water in it " it is raining " tha an turadh ann . is the dry-spell in it " it has stopped raining . " also : rinn e turadh san fheasgar . made it a dry-spell in the afternoon " it stopped raining (for a while) in the afternoon . kevin donnelly	0	ham	Ling spam dataset
Current understanding is #COVID19 spreads mostly from person to person through respiratory droplets produced when a person coughs or sneezes similar to how flu spreads. Learn more at https://t.co/VvIzx7O3mM https://t.co/MiHHHyCfTa	0	ham	Twitter FND dataset
Birx says COVID-19 outbreak not under control because "people are on the move"™ https://t.co/YFqi6f7Rvj	1	spam	Twitter FND dataset

3.4.2 Dataset Balancing

Machine learning is challenged by imbalanced dataset classification. Traditional supervised learning techniques bias toward the majority class in imbalanced datasets, leading to poor classification results for the minority class [230], [238], [239]. Thus, solving the class imbalance problem is essential before applying any ML model. To know the class imbalance for binary-class datasets, the class imbalance ratio (CIR) for datasets is computed as (Size of minority class) / (Size of majority class). The summary of the datasets used, along with their CIR, is given in Table 3.2.

Table 3.2: Summary of individual datasets and *combined dataset* along with CIR

Datasets	Reference	Year	Total instances	#Ham(0)	Ham (%)	#Spam(1)	Spam (%)	CIR
D1 (<i>SMS spam</i>)	[199]	2016	5574	4827	86.59	747	13.40	0.15
D2 (<i>Ling spam</i>)	[240]	2019	2892	2411	83.36	481	16.32	0.19
D3 (<i>Fake news</i>)	[241]	2021	8560	4480	52.34	4080	47.66	0.91
<i>Combined dataset (D1+D2+D3)</i>			17024	11716	68.82	5308	31.17	0.45

Table 3.2 shows an imbalanced distribution of *spam* and *ham* messages in the *SMS spam*, *Ling spam*, and *combined dataset* that must be balanced. The dataset balancing is essential to avoid biased model training, i.e., training on the majority class. The data resampling techniques results in a significant performance improvement for some ML models by altering the composition of the training dataset on imbalanced data. The dataset can be balanced using under-sampling, over-sampling, and hybrid approaches that use both under-sampling and over-sampling approaches [230]. The concept behind the under-sampling technique is to lessen the number of samples of class containing more data equivalent to the class containing the minimum samples, whereas the over-sampling technique is just the inverse of the under-sampling technique that adds more data to the least sample containing class [236].

The following data resampling techniques are used for balancing datasets:

- **NearMiss (NM):** We have used the NearMiss under-sampling technique from the ‘*Imblearn*’ package for balancing using under-sampling.
- **SmoteTomek (ST):** The SmoteTomek technique performs both under-sampling and over-sampling. The ST technique from the ‘*Imblearn.combine*’ package combines two techniques. The SMOTE (Synthetic Minority Over-sampling Technique), built on the KNN algorithm, is used for upsampling, creating synthetic minority class data points to balance the dataset. In contrast, Tomek is a heuristic approach that performs down-sampling. Tomek links classify all the pairs of data points nearest to each other but belonging to different classes (Wang et al., 2019).

However, they also have limitations in addition to the benefits of under-sampling and over-sampling techniques. Under-sampling approaches entail the loss of majority class data points to balance the dataset. In contrast, over-sampling techniques elude the limitation of under-sampling but produce additional samples inside the minority class, resulting in model overfitting. Table 3.3 shows the original dataset shape and the dataset shape after balancing using NM and ST techniques.

Table 3.3: Shape of imbalanced datasets before and after using data balancing techniques

Dataset	Class	Original dataset shape	Resampled shape retrieved using NM technique	Resampled shape retrieved using ST technique
<i>SMS Spam dataset</i>	<i>Ham (0)</i>	4827	747	4825
	<i>Spam(1)</i>	747	747	4825
<i>Ling Spam dataset</i>	<i>Ham (0)</i>	2411	481	2410
	<i>Spam(1)</i>	481	481	2410
<i>Combined Dataset</i>	<i>Ham (0)</i>	11716	5308	11714
	<i>Spam(1)</i>	5308	5308	11714

The resampled dataset shape of *SMS spam* and *Ling spam* retrieved using NM consists of small instances. Hence are ignored for ML and DL-based experimentation. However, the balanced *combined dataset* has comparatively large instances from NM under-sampling technique. Thus, the balanced *SMS spam dataset*, *Ling spam dataset*, and *combined dataset* using the ST technique, along with the balanced *combined dataset* retrieved using the NM technique, are considered to train and test several ML models, as shown in Figure 3.6. However, it's observed that some ML models outperformed on balanced data using the ST technique. The performance comparison of various ML models on imbalanced and balanced datasets is summarized in Section 4 to comprehend how dataset balancing affects ML models performance.

3.4.3 Dataset Pre-processing

Data cleaning and data pre-processing are the pragmatic steps required for any NLP-based text classification. In our experimentation, the *Fake news dataset* is directly passed to the dataset pre-processing phase as it is already balanced. However, the

imbalanced datasets (*SMS spam*, *Ling spam*, and *combined datasets*) are balanced using dataset balancing techniques and then sent to data pre-processing before feeding into any ML/DL model. The Dataset pre-processing involves the following steps:

- **Dataset cleaning:** The datasets were cleaned for any noise (white spaces, newlines, missing values), punctuation marks, special characters, HTML tags, etc., using the *'re'* python library. The shortened words with apostrophes were commonly used in datasets and were corrected using the *contractions* python module.
- **Dataset normalization and standardization:** The cleaned data are converted into lowercase tokens, as some NLP libraries are case-sensitive. The irrelevant data, such as duplicate words and stop words, were removed, which will reduce the actual size of the data. Input features are converted into numerical tokens using the Keras tokenizer, followed by padding to create a dictionary of unique words. Then, tokens are converted into standard form by applying stemming and lemmatization. Lemmatization eliminates morphological information from words, whereas stemming removes prefixes and suffixes. We have used snowball stemmer for stemming and WordNetLemmatizer for lemmatization. The lower the noise, the higher the prediction performance of the models. The categorical output values (*spam/fake* and *ham/real*) are converted into numerical form using the label encoder module of the *Scikit-learn* library.
- **Data split:** Further, the processed data is distributed into two sets, i.e., training data 80% and testing data 20%.

3.4.4 Exploratory Data Analysis

EDA is a process of analyzing data to find patterns, statistics, and graphical representations. The following EDA has been applied to the *combined dataset* to draw some relevant insights:

- **Word Clouds:** The word clouds formed from the *ham* and *spam* words is visualized using *wordcloud()*. The most common stop words, such as “of”, “the”, “a”, “is” etc., are removed before plotting word clouds. The *ham* message word cloud shows that “language”, “system”, “english”, “linguistic” and “people” are the most frequent words

in *ham* messages, as shown in Figure 3.2. The spam message Word cloud shows that “free”, “https”, “CO”, “claim” and “bulk email” are the most commonly used words in spam messages shown in Figure 3.3.



Figure 3.2: Word Cloud for *ham* words



Figure 3.3: Word Cloud for *spam* words

- **Frequency of *Ham* and *Spam* messages over the message length:** From Figure 3.4, it is clear that most of the *ham* messages are of shorter length than spam messages length.

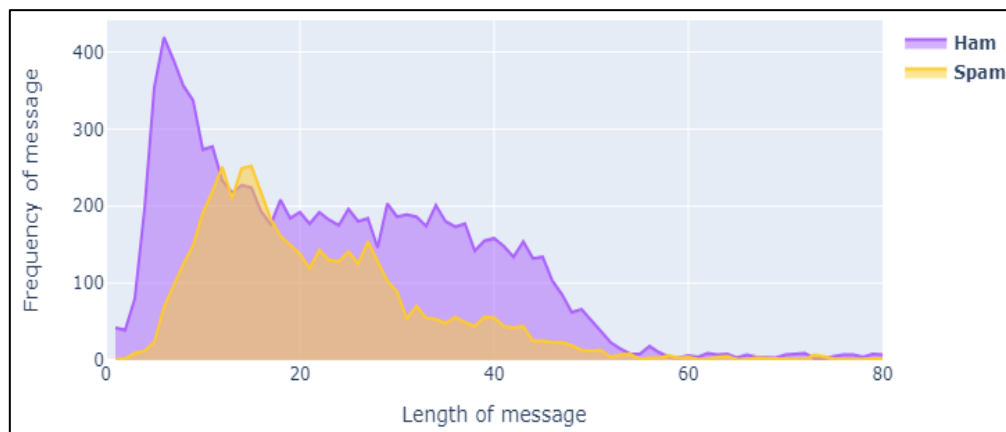


Figure 3.4: Frequency of *ham* and *spam* messages over the length of the message

- **Top 20 most common words in text messages:** Figure 3.5 shows the top 20 most common words used.

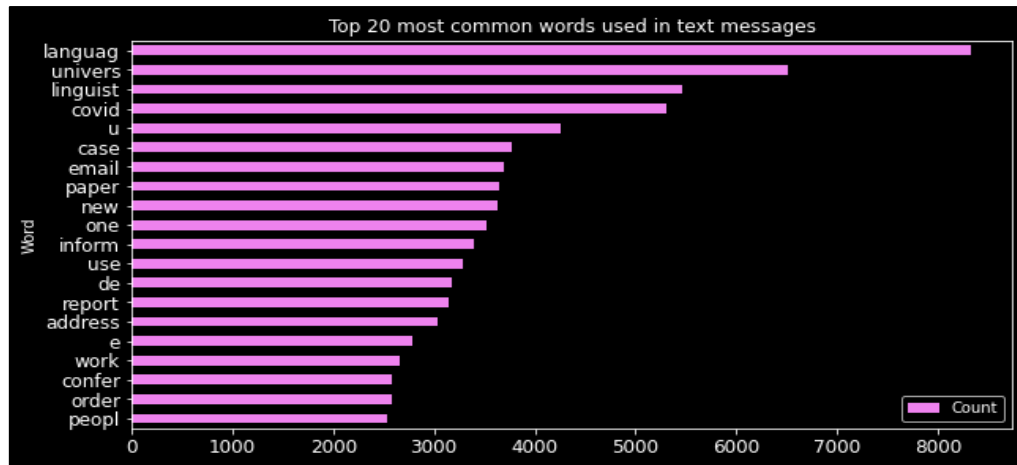


Figure 3.5: Top 20 most common words used in text messages

- **Dataset statistics:** The *describe()* method from the pandas library offers more statistics about the *combined dataset*, such as the count, mean, standard deviation (std), min, and max length of *spam* and *ham* messages. The dataset contains 17024 messages with 16584 unique messages and 440 duplicate messages. The most frequent *ham* message is ‘Sorry, I’ll call later’ with a count of 30, whereas the most frequent *spam* message is ‘this is not spam; you are receiving this mess...’. Table 3.4 describes the statistics for the *combined dataset*.

Table 3.4: Statistics for the *combined dataset*

Lab el	Count	Uniqu e	Top frequent message	Frequenc y	Mean	Std	M in	25%	50%	75%	Max
Ham (0)	11716	11395	Sorry, I'll call later	30	736.2 7	1889.71	2. 0	65.7 5	162. 0	291. 0	2864 9
Spa m (1)	5308	5189	this is not spam; you are receiving this mess..	5	460.6 4	1855.77	6. 0	91.0 0	135. 0	195. 0	2857 1

3.4.5 Baseline Machine Learning Models

The Related Work section shows that most past studies have used basic SL, USL, SSL, EL, and MoA for social spam detection. In our ML-based experimentation, we have selected the best ML models from varied approaches such as Bayesian learning, decision-tree-based, bagging, and boosting from the *scikit-learn* python library. The ML models used BoW and TF-IDF feature representation techniques. The ML models used for the experimentation are as follows:

- **Multinomial Naïve Bayes (MNB):** MNB is based on the Bayesian learning approach to categorize discrete features [138], [142], [147], [148], [202].
- **Support Vector Machine (SVM):** The linear support vector classification (LSVC) model from SVM provides the best-fit hyperplane to categorize the input data [138], [157], [202]. Some parameter values are changed for probability estimates.
- **K nearest neighbors (KNN):** KNN uses closeness to categorize into classes [131], [138], [147].
- **Random Forest (RF):** RF model constructs decision trees on samples and uses the majority vote to classify [138], [148], [157], [158], [202].
- **Bagging classifier (BC):** BC fits base classifiers on random subsets of the original dataset and then combines (votes or averages) their separate predictions to make a final prediction.
- **ExtraTree classifier (ETC):** ETC is a meta-estimator that uses randomized/extra decision trees on subsamples of the dataset to boost accuracy and control overfitting by aggregating the prediction of the individual decision tree.
- **Gradient Boosting classifier (GBC):** GBC is based on boosting technique that iteratively learns from weak classifiers to construct a robust model by leveraging an ensemble of weak classifiers [142], [148], [157], [158].

- **Extreme Gradient Boosting (XGB):** XGB is a popular boosting model based on a regularizing gradient boosting approach. It generates an ensemble of weak prediction models, generally decision trees, for classification stage-by-stage. However, allows the optimization of an arbitrary loss function [90].
- **AdaBoost classifier (ADB):** Adaboost, called Adaptive boost, is based on an ensemble approach. The weights are reallocated to each instance, with higher weights assigned to incorrectly classified instances.

The aforementioned baseline models perform comparatively better than other ML models for detecting text spam. For training baseline ML models, the parameter tuning of ML is done to find the best predictive model. Later the models are tested on testing data (20%) to predict the performance of each ML model. The ensemble models RF, ETC, BC, and XGB provided good results on imbalanced datasets. Yet, the results are significantly improved on balanced datasets. For enhancing the performance score further, the prediction result of each ML model is used in our proposed soft and hard voting ensemble models, i.e., PHVE and PSVE, as described in the subsequent subsection.

3.4.6 Proposed PHVE and PSVE Models

To get a better combined prediction from multiple classifiers, generally bagging, boosting, and stacking/voting ensembles are acknowledged by the research community. In our experimentation, we considered the voting classifier provided by the ensemble module of the *sklearn* library. The voting ensemble model combines the performances of multiple models to make predictions using a list of estimators as arguments and a voting method. After applying the ML models individually, the predictions from individual models are combined in two ensemble models, namely the proposed hard voting ensemble (PHVE) and the proposed soft voting ensemble (PSVE), to get the most accurate prediction results. Figure 3.6 depicts the pictorial representation of ML-based proposed hard voting ensemble (PHVE) and soft voting ensemble (PSVE) models.

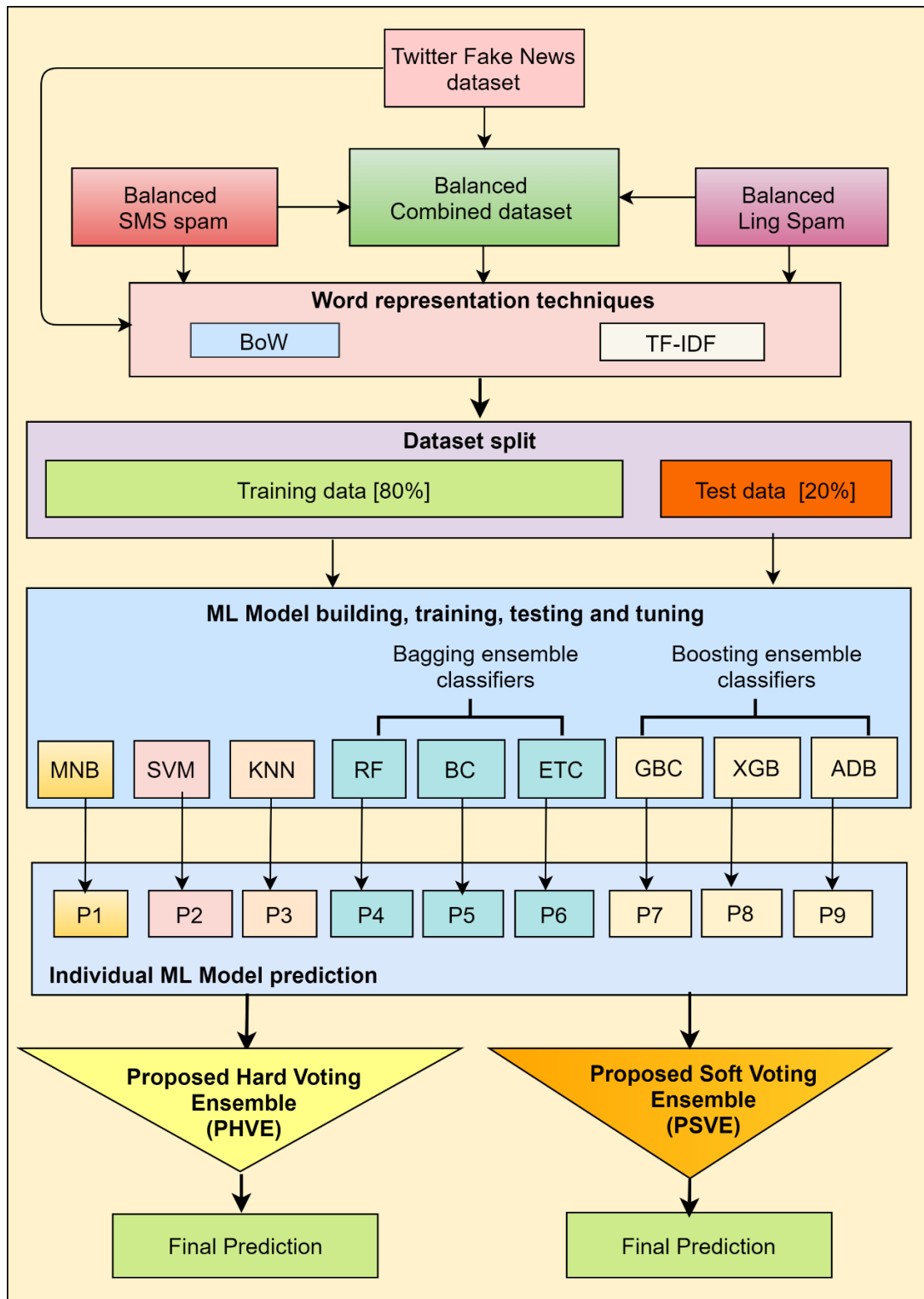


Figure 3.6: Schematic for proposed hard voting ensemble (PHVE) and soft voting ensemble (PSVE) models

Proposed hard voting ensemble (PHVE): The PHVE model is based on hard voting that uses majority voting. The predicted output class has the most votes of being predicted by each classifier as described in Eq. (5),

$$\hat{y}_h = \text{mode}\{C_1(x), C_2(x), \dots, C_m(x)\} \quad (5)$$

where C_j is classifier, $\hat{y}_h$ predicts the final class label via majority voting of each classifier C_j .

In the PHVE model, an ensemble of nine ML models, i.e., MNB, SVM, RF, XGB, ETC, BC, KNN, ADB, and GDB, is built by passing the list of classifiers in the ‘*estimator*’ parameter. The voting parameter is selected as ‘*hard*’. Each ML model is applied with uniform weight using the ‘*weights*’ parameter. Then the final class prediction is given by the mode of each classifier’s class prediction. The predicted class label for a particular sample is the class label that signifies the majority (mode) of the class labels predicted by each classifier. For example, if there are 3 classifiers- *clf1*, *clf2*, *clf3* for specific data with their predictions as (0,1,1), then for uniform weights assigned to the classifiers, the mode of the predictions is taken. Thus, the mode of (0,1,1) is 1. Hence, the predicted class to the particular data becomes *class 1*. The various parameter settings for building and training PHVE are provided in Table 9. Later, the performance of the PHVE model is validated on the testing dataset. After the model evaluation on test data, the best parameter values are extracted, and the best prediction model is finalized by adjusting the parameters. In PHVE, the final class prediction is made using the majority voting of each classifier.

Proposed soft voting ensemble (PSVE): In PSVE, the predicted probabilities for class labels are summed and predicted with the class label with the largest sum probability. In contrast to PHVE, PSVE returns the class label as argmax of the sum of predicted probabilities. Mathematically, the soft voting mechanism ($\hat{y}_s$) can be expressed as Eq. (6),

$$\hat{y}_s = \underset{i}{\operatorname{argmax}} \sum_{j=1}^m w_j p_{ij} \quad (6)$$

where w_j is the weight of j th classifier, $i \in \{0, 1\}$ for binary classification, p_{ij} is the

predicted probability of the j th classifier for class label i .

Similar to PHVE, in the PSVE model, an ensemble of nine ML models, i.e., MNB, SVM, RF, XGB, ETC, BC, KNN, ADB, and GDB, is built by passing the list of classifiers in the estimator parameter. The voting method is selected as ‘soft’. Each ML model is applied with uniform weight using the ‘weights’ parameter. The predicted class probabilities for each classifier are collected, multiplied by the classifier weight, and averaged. The class label with the highest average probability is then used to determine the final class label. For example, if there are three binary classifiers- $clf1$, $clf2$, and $clf3$. For a particular data, the classifiers make the following predictions in terms of probabilities in favor of classes $(0,1)$ as $clf1$ (0.2, 0.8), $clf2$ (0.1, 0.9), and $clf3$ (0.8, 0.5). With uniform weights, the average probability of $class\ 0 = (0.2 + 0.1 + 0.8)/3 = 0.366$ and the average probability of $class\ 1 = (0.8 + 0.9 + 0.5)/3 = 0.733$. Then, the average probability predicted by the ensemble classifier for both classes will be [36.6%, 73.3%]. If the threshold is 0.5, the final predicted $class\ \hat{y}_s = \underset{i}{argmax} [36.6\%, 73.3\%] = 1$, i.e., most likely be predicted with the class label with the largest sum probability. The various parameter settings for building and training PSVE are provided in Table 9. Later, the performance of the PSVE model is validated on the testing dataset. After the model evaluation on test data, the best parameter values are extracted, and the best prediction model is finalized by adjusting the parameters. The target label with the highest sum of weighted probability is chosen as the final prediction. Finally, Section 4.4 presents the performance evaluation of PHVE and PSVE models compared to other ML baseline models on various imbalanced and balanced datasets.

3.5 Experimental Setup and Results

3.5.1 Dataset Description

Due to a lack of a bigger social spam corpus, the *combined dataset* is merged from *SMS Spam*, *Ling Spam*, and *Fake news datasets*. The *combined dataset* is imbalanced. Thus, NM and ST are used to balance it. The balanced datasets for ML and DL experimentation are split into training data (80%) and testing data (20%).

3.5.2 Experiment Setting

The following sub-sections give information about experiment settings and optimized hyperparameters used for ML experimentation.

The ML experiments are performed with python 3.7. The GPU used for experimentation is Tesla T4. Parameter setting is done for individual ML models and proposed PHVE and PSVE models as described in Table 3.5.

Table 3.5: Parameter values for ML models

ML Classifier	Parameter value
MNB, KNN, RF, ETC, BC, GDB, ADB, XGB	Default
SVM (SVC)	probability=True, random_state=None, kernel='rbf', shrinking=True, tol=0.001, verbose=False
PHVE (Proposed)	(estimators=[('MNB', clf1), ('SVM', clf2), ('KNN', clf3), ('RF', clf4), ('ETC', clf5), ('BC', clf6), ('GDB', clf7), ('ADB', clf8), ('XGB', clf9)], voting='hard', weights=None, n_jobs=None, flatten_transform=True, verbose=False)
PSVE (Proposed)	(estimators=[('MNB', clf1), ('SVM', clf2), ('KNN', clf3), ('RF', clf4), ('ETC', clf5), ('BC', clf6), ('GDB', clf7), ('ADB', clf8), ('XGB', clf9)], voting='soft', weights=None, n_jobs=None, flatten_transform=True, verbose=False)

3.5.4 Results and Discussions

The performance comparison of nine individual ML models with PHVE and PSVE is conducted on imbalanced and balanced datasets to evaluate social spam detection. The variation in performance of various ML/DL models on imbalanced and balanced datasets is evaluated using Accuracy variation (AV). It is the difference between ML/DL model's accuracy obtained on balanced datasets using the ST data resampling approach and the accuracy obtained on the imbalanced datasets. The top five best performance scores from each dataset are highlighted in Table 3.6-3.9.

Table 3.6 provides the performance comparison of nine ML baselines with PHVE and PSVE on imbalanced *SMS spam* and balanced *SMS spam dataset* using the ST

technique. The ML baseline models, PHVE and PSVE, improve on the balanced *SMS spam dataset*. Our proposed soft voting ensemble (PSVE) model achieves the highest accuracy of 98.21% on a balanced *SMS spam dataset*. Some other ML models with good performance on balanced *SMS spam dataset* are PHVE, ETC, SVM, and XGB. The accuracy variation (AV) of the KNN model is maximum, i.e., 8.24%, and minimum with the RF model, i.e., 1.14%. Other models accuracy is also improved with a variation of approximately (1~ 4) %.

Table 3.7 provides the performance comparison of various ML baselines with PHVE and PSVE on imbalanced *Ling spam* and balanced *Ling spam datasets* using the ST technique. ML baseline models, PHVE and PSVE, improve on the balanced *Ling spam dataset*. The PHVE model achieves the highest accuracy of 93.09% on the balanced *Ling spam dataset*. Some other ML models with good performance on balanced *Ling spam dataset* include PSVE, ETC, SVM, and RF. From the AV of Table 3.7, it is observed that the accuracy of the KNN model is decreased by 21.39%, and accuracy for other ML models is improved within a range of (1~4)% in the balanced *Ling spam dataset*.

The *Twitter Fake news dataset* is already balanced. Thus, a performance comparison of various ML baselines with PHVE and PSVE, only on the balanced *Fake news dataset*, is provided in Table 3.8. Our proposed soft voting ensemble (PSVE) model achieves the highest accuracy of 95.76% and ROC-AUC score of 0.98 on the *Twitter Fake news dataset*. Some other ML models with good performance on the *Twitter Fake news dataset* include PHVE, ETC, SVM, and RF.

Table 3.6: Performance comparison of nine ML baselines with proposed voting ensemble models on imbalanced and balanced *SMS spam dataset*

	Imbalanced SMS spam dataset					Balanced SMS spam dataset retrieved using ST					~AV (%)
Performance parameters→ ML Models ↓	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	
MNB	89.36	89.34	89.36	89.34	0.93	93.29	93.32	93.29	93.29	0.96	3.93
SVM	92.04	92.11	92.04	92.04	0.94	95.07	95.16	95.07	95.07	0.98	3.03
RF	93.04	93.09	93.04	92.96	0.96	94.18	94.24	94.19	94.18	0.98	1.14
XGB	91.52	91.55	91.52	91.52	0.95	94.70	94.71	94.70	94.70	0.98	3.18
ADB	90.43	90.44	90.43	90.42	0.94	93.60	93.48	93.60	93.45	0.96	3.17
BC	89.64	89.65	89.64	89.54	0.93	92.23	92.17	92.23	92.18	0.95	2.59
ETC	93.69	93.74	93.71	93.68	0.96	97.54	97.66	97.64	97.34	1.00	3.85
GDB	89.23	89.23	89.23	89.21	0.93	91.08	91.15	91.08	91.06	0.94	1.85
KNN	66.82	66.85	66.81	66.82	0.80	75.06	75.06	75.08	75.06	0.85	8.24
PHVE	93.83	93.85	93.84	93.76	0.99	97.87	97.89	97.87	97.35	1.00	4.04
PSVE	94.28	94.32	94.28	94.25	1.00	98.21	98.25	98.22	98.14	1.00	3.93

Table 3.7: Performance comparison of nine ML baselines with proposed voting ensemble models on imbalanced and balanced *Ling spam dataset*

	Imbalanced <i>Ling spam dataset</i>					Balanced <i>Ling spam dataset</i> retrieved using ST					~AV (%)
Performance parameters→ ML Models ↓	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	
MNB	80.61	80.67	80.61	79.96	0.92	85.55	85.59	85.55	85.54	0.95	4.94
SVM	89.01	89.04	89.01	89.02	0.96	91.68	91.70	91.68	91.66	0.97	2.67
RF	88.58	88.65	88.58	88.50	0.96	90.88	90.90	90.88	90.85	0.97	2.30
XGB	87.74	87.78	87.74	87.75	0.96	89.69	89.70	89.69	89.69	0.96	2.45
ADB	81.91	81.92	81.91	81.90	0.92	83.48	83.54	83.48	83.24	0.94	1.57
BC	85.52	85.55	85.51	85.51	0.96	87.89	87.90	87.89	87.65	0.96	2.37
ETC	91.93	91.95	91.93	91.90	0.97	95.01	94.98	95.01	95.01	0.97	3.08
GDB	81.75	81.76	81.75	81.73	0.92	83.06	83.08	83.06	83.06	0.94	1.31
KNN	90.70	90.74	90.70	89.98	0.96	69.31	79.02	69.31	69.38	0.92	-21.39
PHVE	91.65	91.67	91.65	91.63	0.96	93.09	93.10	93.09	92.99	0.98	1.44
PSVE	88.97	88.98	88.97	88.92	0.97	90.05	90.08	90.05	90.04	0.97	1.08

Table 3.8: Performance comparison of nine ML baselines with proposed voting ensemble models on balanced *Fake news dataset*

Balanced Twitter spam dataset					
Performance parameters→ ML Models ↓	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score
MNB	90.36	90.36	90.36	90.36	0.89
SVM	93.44	93.42	93.44	93.40	0.95
RF	92.31	92.32	92.31	92.28	0.94
XGB	89.67	89.67	89.67	89.67	0.93
ADB	86.84	86.84	86.84	86.84	0.90
BC	88.86	88.86	88.86	88.86	0.90
ETC	95.57	95.57	95.57	95.57	0.98
GDB	89.46	89.46	89.46	89.46	0.91
KNN	81.90	81.90	81.90	81.90	0.90
PHVE	93.85	93.82	93.86	93.87	0.96
PSVE	95.76	95.76	95.76	95.76	0.98

As mentioned, the *combined dataset* is balanced using two data resampling techniques. Thus performance comparison of various ML baselines with PHVE and PSVE is done on the imbalanced *combined dataset*, balanced *combined dataset* retrieved using NM, and balanced *combined dataset* retrieved using ST techniques is provided in Table 3.9. The ML baseline models, except KNN, improve on the balanced *combined dataset* retrieved using NM and on the balanced *combined dataset* retrieved using the ST technique. ML models' accuracy is improved within a range of (3 ~ 9) % on the balanced *combined dataset* retrieved using the ST technique. The accuracy of the KNN model is decreased by 14.38% on the balanced *combined dataset* retrieved using the ST technique. The PSVE model achieves the highest accuracy of 95.88% with a 0.98 ROC-AUC score on a balanced *combined dataset* retrieved using the ST technique. The PHVE model also performed well, with an accuracy of 94.46% on the balanced *combined dataset* retrieved using ST. Furthermore, the analysis reveals that the ETC, SVM, RF, XGB, and BC models also produced good results and contributed considerably to both PHVE and PSVE.

The consolidated accuracy analysis is carried out graphically by comparing PHVE and PSVE with other good-performing ML models in terms of their accuracy on different imbalanced and balanced datasets, as shown in Figure 3.7. The ML models with 88% or above accuracy are considered good and compared graphically with PHVE and PSVE. The performance of PHVE and PSVE models are highlighted with data labels to have a quick comparison with other ML models on imbalanced and balanced datasets. The PSVE model outperformed other models on almost all datasets except the imbalanced *Ling spam*, balanced *Ling spam*, and imbalanced *combined dataset*. The performance of the PHVE model is better on imbalanced *Ling spam* and balanced *Ling spam*. PHVE and PSVE perform better when baseline ML models improve. The major performance improvements in baseline models, PHVE and PSVE, are observed specifically on the balanced *SMS spam* and balanced *combined dataset* using the ST technique than on imbalanced datasets. The KNN model's performance is relatively lower, but it produced better accuracy of 90.70% on the imbalanced *Ling spam dataset*. The PHVE and PSVE provide considerably good results even if some baseline models are not performing well. The drawback of PHVE, PSVE, and SVM is that they took longer to train and test models than other baseline ML models.

Table 3.9: Performance comparison of various ML baselines with proposed voting ensemble models on the imbalanced *combined dataset*, balanced *combined dataset* retrieved using NM and ST data sampling technique

Performance parameters→ ML Models ↓	Imbalanced Combined dataset					Balanced Combined dataset retrieved using NM					Balanced Combined dataset retrieved using ST					~AV (%)
	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	
MNB	82.23	82.26	82.23	82.27	0.94	85.40	85.42	85.40	85.41	0.96	91.66	91.68	91.66	91.65	0.96	9.43
SVM	90.10	90.17	90.12	90.16	0.96	91.16	92.57	92.56	92.50	0.97	95.13	95.15	95.13	95.14	0.98	5.03
RF	87.27	87.24	87.27	87.25	0.95	91.24	91.25	91.24	91.23	0.97	93.78	93.80	93.78	93.77	0.99	6.51
XGB	83.48	83.51	83.48	83.52	0.92	86.07	86.09	86.07	86.07	0.96	91.83	91.84	91.83	91.84	0.98	8.35
ADB	83.97	83.94	83.97	83.92	0.92	84.04	84.05	84.04	84.04	0.95	88.32	88.31	88.32	88.30	0.93	4.35
BC	84.66	84.67	84.66	84.64	0.92	85.17	85.11	85.17	85.12	0.96	92.07	92.05	92.07	92.04	0.98	7.41
ETC	91.59	91.55	91.59	91.54	0.96	92.10	92.07	92.10	91.96	0.98	95.18	95.20	95.18	95.13	0.99	3.59
GDB	79.87	79.83	79.87	79.85	0.93	80.79	80.77	80.79	80.73	0.94	86.28	86.25	86.28	86.27	0.95	6.41
KNN	73.30	73.36	73.30	73.34	0.69	55.74	55.72	55.74	55.74	0.67	58.95	58.92	58.95	58.98	0.70	-14.38
PHVE	88.79	88.81	88.79	88.80	0.96	91.49	91.50	91.49	91.48	0.98	94.46	94.42	94.46	94.48	0.99	5.67
PSVE	89.23	89.29	89.23	89.24	0.96	92.47	91.45	91.47	91.50	0.98	95.88	95.70	95.88	95.84	0.99	6.65

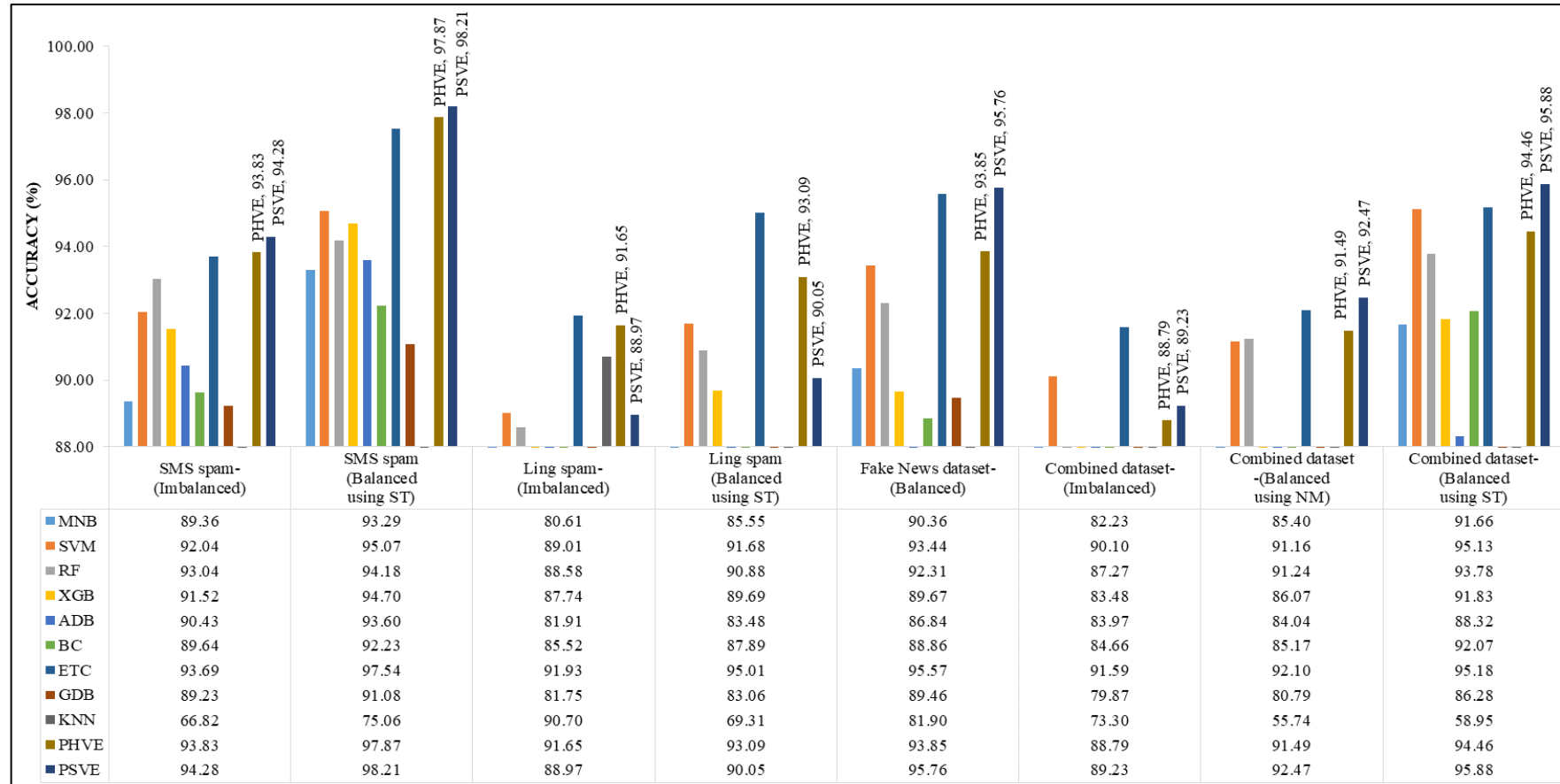


Figure 3.7: Consolidated accuracy analysis of proposed ensemble models (PHVE and PSVE) with other good-performing ML models on different imbalanced and balanced datasets.

3.5.5 Comparison with Existing Studies

The accuracy comparison of proposed ensemble models, i.e., PHVE and PSVE, with other existing ML-based studies/approaches is given in Table 3.10.

Table 3.10: Performance comparison of *PHVE* and *PSVE* with existing ML-based approaches

References	Feature extraction/ Optimization technique	ML/DL Classifier	Dataset/ Balancing technique	A (%)
(Gupta et al., 2018)	BoW	SVM, NB, RF, and GB algorithms	<i>Hspam14 dataset</i>	91.65
[242]	PSO	NB	<i>Ling spam</i>	95.50
[233]	CV, TF-IDF	MNB, Neural Network, LR and SVM	<i>Twitter spam dataset</i>	93.02
PHVE (Proposed)	BoW, TF-IDF	MNB, SVM, RF,	<i>SMS spam</i>	97.87
		XGB, ADB, BC,	<i>Ling spam</i>	93.09
		ETC, GDB, KNN,	<i>Twitter FND</i>	93.85
		PHVE	<i>Combined dataset using NM</i>	91.49
			<i>Combined dataset using ST</i>	94.46
PSVE (Proposed)	BoW, TF-IDF	MNB, SVM, RF,	<i>SMS spam</i>	98.21
		XGB, ADB, BC,	<i>Ling spam</i>	90.05
		ETC, GDB, KNN,	<i>Twitter FND</i>	95.76
		PSVE	<i>Combined dataset using NM</i>	92.47
			<i>Combined dataset using ST</i>	95.88

3.6 Summary

Addressing OSN spam is crucial. Existing ML-based techniques are ineffective due to class imbalance concerns, lack of bigger labeled datasets, production of relevant features, performance challenges, spam drift, and more. Small and relatively imbalanced datasets have been used in several existing research. ML-based ensemble models are presented to tackle the most pressing problems using a larger and more diverse dataset. Initially, the lack of a large labeled dataset is solved by merging the three benchmark datasets to build large *spam* and *ham*-labeled corpus *combined dataset*. Another key issue of the class imbalance problem is solved, as unequal distribution substantially reduces the classifier's performance. The data balancing is done using NearMiss and SmoteTomek data resampling techniques. Subsequently, several pre-processing and word representation techniques are applied to the imbalanced and balanced datasets to prepare them for ML model training. Later, for the ML-based experimentation, the best models from varied approaches are selected, trained, and tested on imbalanced and balanced datasets. The comparative results of different ML models with PHVE and PSVE on various imbalanced and balanced datasets are presented with the five good performance scores highlighted. Among various ML models, our proposed voting ensemble model, i.e., PSVE, outperformed with an accuracy of 95.88% on balanced *combined dataset*. Other good-performing models are PHVE, ETC, SVM, and RF. Most ML models performance increased on balanced datasets retrieved using the ST technique except KNN. The KNN model performed better on the imbalanced *Ling spam dataset* with an accuracy of 90.70% than the balanced *Ling spam dataset*. The performance of proposed voting ensemble models and other baselines considerably improved on balanced *combined dataset* retrieved using the ST technique. The ML models produced accurate and useful results but could not learn low-level features, which is not a problem for DL models. DL-based approaches are found to be robust and effective. Therefore, DL-based experimentation is employed on the *combined dataset* retrieved using the ST technique in the subsequent chapter to further improve social spam detection performance.

Chapter 4 : Proposed GCB-SA and FCB-SA approaches using Deep Learning

4.1 Introduction

The proposed voting-based ML models in our social spam detection framework provided satisfactory results and outperformed baseline ML models. The accuracy of ML models is observed to be improved on balanced datasets retrieved using the ST data balancing technique. In ML-based approaches, traditional frequency-based word representation techniques such as BoW and TF-IDF are used to generate feature vectors. However, such word representation techniques are time-consuming and inefficient in producing contextual word vectors. Moreover, they cannot learn from low-level features, which is not an issue with recent word embeddings for DL-based approaches. DL models are used to learn tasks from another dataset using the pre-trained model that need not be trained from abrasion, thus significantly improving the model's performance by reducing training time. Also, pre-trained word embeddings and deep learning models have shown promising results in text classification.

This chapter presents hybrid DL-based approaches in the framework for social spam detection to further improve the performance by leveraging advanced pre-trained word embedding techniques, deep learning approaches, and self-attention mechanism on a balanced *combined dataset* retrieved using the SmoteTomek data sampling technique. The DL-based approaches are experimented with embeddings generated using GloVe and FastText pre-trained word embeddings.

The major findings of this chapter have been peer-reviewed and published in:

- Rao, S., Verma, A. K., & Bhatia, T. (2023). Hybrid Ensemble Framework with Self-Attention Mechanism for Social Spam Detection on Imbalanced Data. *Expert Systems with Applications*, Elsevier, 217, 119594. [I.F. 8.665]. SCIE.

In the proposed deep learning-based hybrid approaches, generated embeddings are passed into the sequential model comprised of Conv1D and Bi-directional recurrent neural network layers with the self-attention mechanism for improved context understanding and effective results. This chapter examines DL-based approaches for detecting social spam using word embedding techniques, hyper-parameter optimization, and a self-attention mechanism. Experiments and comparisons with other techniques show that the proposed hybrid framework with deep learning-based approaches achieves better performance.

4.2 Related Work

Some studies have improved spammer detection systems using word embedding, optimization, and DL techniques. DL-based frameworks begin with word embedding. Low-dimensional vectors represent words in word embedding by understanding syntax and semantics. Words with similar meanings usually appear in similar contexts [243].

Feng et al. [148] developed a multistage framework that uses the Word2Vec model for identifying spam messages and malicious URLs in Sina-Weibo OSN. Madisetty & Desarkar [244] have built a CNN-based ensemble with two-word embeddings (Word2Vec and GloVe) to construct a feature-based model on balanced HSpam14 [113] and imbalanced 1KS10KN Tweets datasets.

Jain et al. [147] have developed a Semantic Convolutional Neural Network (SCNN) approach that leverages features from Word2Vec, wordNet, and conceptNet word embeddings. Their SCNN model used the CNN model with a semantic layer on Twitter and *SMS spam* datasets. Later the same authors, Jain et al. [245], proposed CNN and LSTM-based sequential stacked (SSCL) models using the same word embeddings and datasets. Ban et al. [138] have developed a Bi-LSTM approach that leverages DL models, Word2Vec, and statistical features from the Twitter text.

Barushka & Hajek [246] implemented a framework that used TF-IDF for feature extraction and a multi-objective evolutionary feature selection (MOEFS) with a regularized deep

neural network (RDNN), to minimize the classifier miss-classification rate on two datasets, Hyves and Twitter dataset.

Lai et al. [202] have built a neural network-based system using CNN, LSTM, TF-IDF, and Word2Vec to detect fake news. ML models require a lot of data to build good NLP models. Extracting relevant features from large data for certain domains or languages becomes tedious and costly. Transfer learning methods are possible in such situations. Transfer learning models leverage pre-trained models such as ResNet50, GloVe, and DenseNet to rebuild and reuse DL models [87], [229].

4.3 Contribution

The following contributions have been made to determine the most efficient approaches for detecting social spam:

- Larger and more diverse spam corpus (*combined dataset*) is created by merging three benchmark datasets: SMS spam, Ling spam, and Twitter fake news.
- DL-based experimentation is done on a balanced *combined dataset* retrieved using the ST data sampling technique.
- Embedding layer is generated using GloVe and FastText pre-trained word embedding techniques.
- Proposed GCB-SA and FCB-SA hybrid approaches for better contextual output and efficient results over a balanced *combined dataset*.
- In the GCB-SA approach, GloVe pre-trained word embeddings are passed (Conv1D+Bi-GRU+Bi-LSTM) sequential model with the self-attention mechanism, whereas in the FCB-SA FastText embeddings are given to (Conv1D+Bi-GRU+Bi-LSTM) sequential model with the self-attention mechanism.
- Performance comparison of proposed hybrid DL approaches (GCB-SA and FCB-SA) is done with various baseline models using various performance measures.
- Finally, the performance of proposed hybrid DL approaches (GCB-SA and FCB-

SA) is compared with existing approaches/frameworks.

4.4 Proposed Methodology

The overall proposed framework with ML and DL approaches for social spam detection to categorize the text message as *Spam* and *Ham* is depicted in Figure 3.1. This chapter presents the proposed hybrid DL-based approaches (GCB-SA and FCB-SA) for social spam detection.

ML models performed comparatively better on the balanced datasets using the ST data sampling technique. Thus, we have considered only the balanced *combined dataset* built using the ST technique for DL-based experimentation. The pre-trained word embeddings used for DL-based approaches are GloVe and FastText. Facebook's FastText pre-trained word embedding is new, fast, and accurate in finding rare and out-of-vocabulary (OOV) words. The pre-trained word embeddings are loaded to convert input data into tokens and generate word index vectors. For generating same-length text sequences of word vectors, padding is done. Later, various baseline and proposed DL-based neural network architectures are trained using the generated embedding matrix from pre-trained word embeddings. The following subsections describe various baseline DL approaches and proposed hybrid DL approaches for social spam detection.

4.4.1 Baseline DL Models

The following DL-based baseline neural networks are employed with GloVe and FastText word embeddings for the experimentation:

- **Recurrent neural network (RNN):** RNN is used for modeling sequential data. The architecture of RNN uses the current state at time step t as h_t , the previous state at time step t as h_{t-1} , output at time step t as o_t , and input at time step t as x_t . It checks dependency between the words in the text while making predictions but cannot generalize to variable-length sequences due to short-term memory issues. For such problems, Long short-term memory (LSTM), Gated Recurrent Unit (GRU), bi-directional LSTM, and bi-

directional GRU-based gating mechanisms are recommended [88], [247], [248]. The simple architecture of RNN is shown in Figure 4.1.

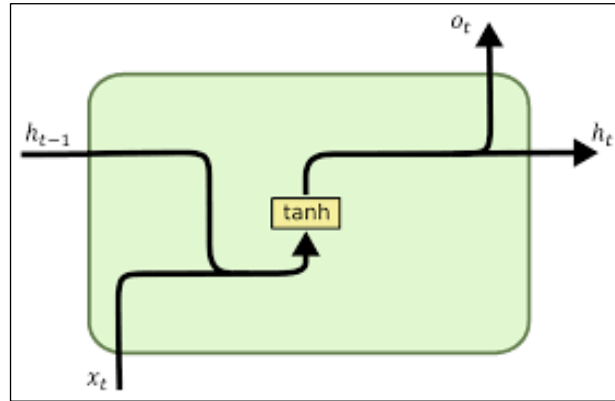


Figure 4.1: Simple architecture of RNN

- **LSTM and Bi-LSTM:** LSTM is a unidirectional RNN that solves the vanishing gradient in backpropagation using a gating mechanism. The architecture of LSTM uses an input gate i_t , forget gate f_t , output gate o_t and cell state c_t . The historical information or output of the previous stage is denoted by h_{t-1} and current input by x_t as shown in Figure 4.2.

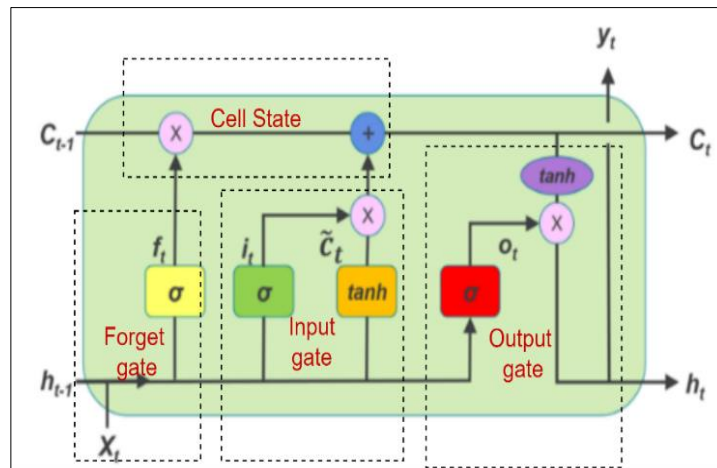


Figure 4.2: Architecture of LSTM

The function of forget gate is to select which information to remove from the cell state and is mathematically described in Eq. (7):

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (7)$$

where W_f is weight matrix, b_f is the bias matrix of forget gate, and σ is the activation function

The output of f_t is 0 if old information is completely discarded. Otherwise, 1. The current cell c_t updates by forgetting some information and inserting some information from the current input x_t and previous hidden state h_{t-1} , which is decided by forget gate f_t , and input gate i_t respectively in each step. The input gate selects what information is stored in a cell using activation functions, the *sigmoid* layer, and the *tanh* layer. The sigmoid layer controls what value to update, and the *tanh* layer help in creating a new candidate value vector $\tilde{C}_t$ as mentioned in Eq. (8), (9), and (10).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (8)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (9)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (10)$$

Although the current hidden state h_t is determined by the present memory cell and output gate o_t . The output gate uses a sigmoid layer to decide which section of the cell state will be the output as described in Eq. (11). Eventually, the *tanh* layer with training matrix weights W_C and bias b_C provides candidate value through the input gate i_t that will be added to the new state of a memory cell C_t along with old state C_{t-1} provided by forget gate f_t . Finally, Eq. (12) represents the hidden state h_t based on the memory cell C_t as mentioned in Eq. (10).

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (11)$$

$$h_t = o_t * \tanh(C_t) \quad (12)$$

Bi-LSTM is an extended version of LSTM that can work in both forward and backward directions to provide better performance than RNNs and LSTMs [230], [245], [248]. The forward LSTM $h_{t (forward)}$ gathers the input sequences of past data information, whereas

the backward LSTM $h_t (backward)$ receives input sequences of future data information. Later the output from both forward and backward layers are combined, as mentioned in Eq. (13).

$$h_t = h_t (forward) \oplus h_t (backward) \quad (13)$$

The LSTM and Bidirectional modules are imported from the sequential module of *Keras.models*. The hyper-parameters used for the Bi-LSTM model are mentioned in Table 4.3.

- **GRU and Bi-GRU:** GRU is an alternative to LSTM proposed by [249] that uses gates to control memorizing. The GRU is faster to train and less complex than LSTMs due to fewer tensor operations. GRU uses two gates: the update gate and the reset gate. It doesn't have an output gate but an update gate that combines the input and forget gate functionality, as shown in Figure 4.3.

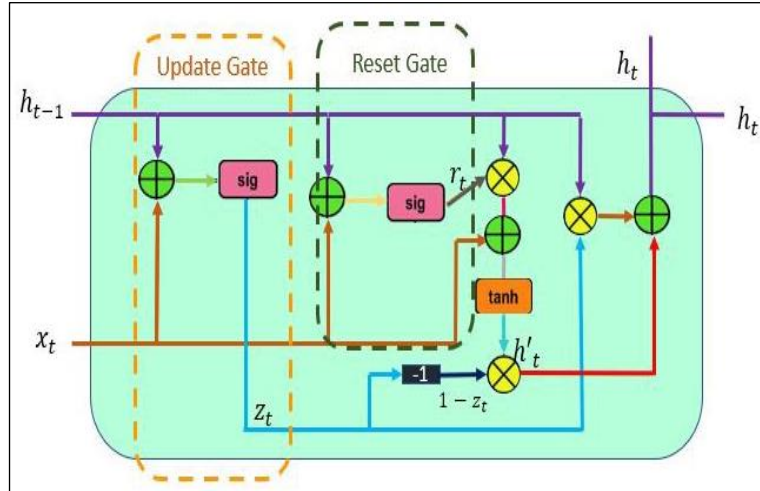


Figure 4.3: Architecture of GRU

The function of the update gate z_t determines the amount of prior information that must be passed to the subsequent state. The function of the reset gate r_t is to determine the amount of past information required to be ignored. The update process of the hidden state h_t at time(t) is described in Eq. (16). Mathematically, the update gate, reset gate, and hidden

state functions used for computation are described in Eqs. (14), (15), and (16), respectively:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (14)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (15)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tanh(W \cdot [r_t * h_{t-1}, x_t]) \quad (16)$$

Similar to Bi-LSTM, Bi-GRU is an extended version of GRU that can work in both forward and backward directions to provide better performance. The GRU and Bidirectional modules are imported from the sequential module of *Keras.models*. The hyper-parameters used for the Bi-GRU model are mentioned in Table 4.3.

- **Convolutional Neural Network (CNN):** RNNs are trained to identify patterns sequentially, whereas CNNs learn to identify patterns across space [250]. CNN-based architectures are useful in text classification to detect patterns of a particular sentiment or topic [248]. The CNN architecture provides convolutional layers with dynamic pooling layers to generate a feature map over the sentence to explicitly capture short and long-range relations of phrases or words. In the proposed DL approaches, one-dimension convolutional neural network (Conv1D) layers are added. The embedding matrix generated from pre-trained word embeddings is initially passed as the sequential model's first layer. The parameters such as vocabulary length, embedding dimensions, and maximum input length of input sequences are computed in the pre-processing and tokenization phases. If the length of the text message is more than the specified maximum length, then it will be truncated; if it is less than the maximum length specified, then zero padding is done. Eventually two Conv1D layers with max-pooling1D layers are added with optimized hyper-parameters as mentioned in Table 4.3.
- **Self-attention (SA) mechanism:** SA is used for improved contextual output and effective results. The attention mechanism dynamically emphasizes some tokens in the input sequence by changing the token embeddings. In the SA mechanism, the multiple inputs are interacted with each other “*self*” to extract the most significant inputs and aggregate their

representations based on attention weights and attention bias to form the output as a sentence vector [251], [252]. With the SA layer, the spam words such as “lottery”, “promotion”, “discount”, “free”, “offer” etc., can be given more attention. This will optimize the feature extraction process, hence increasing the overall performance of the neural network. The SA layer consists of *build()* and *call()* functions. The *build()* function is used to add attention weights and attention bias to build input shape, while the *call()* function is applied using *softmax()* across the attention scores. Later the attention scores received from *softmax()* are multiplied by the corresponding values to generate weighted values. The final output is given by summing the generated weighted values. Mathematically, the SA computation is done as described in Eq. (17):

$$SA(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (17)$$

where Q , K , and V are the query vector matrix, key vector matrix, and value vector matrix, respectively, obtained from input and other parameters.

Initially, the query vector of the current input is multiplied by key vectors from other inputs. Later the score obtained is divided by the scaling factor $(\sqrt{d_k})$. The *softmax()* function normalizes the results into probability distribution on all SA scores and is multiplied by the value vector. Finally, we sum up the weighted value vectors to get the SA output for the word. Subsequently, the same process is applied to all input sequences [252].

4.4.2 Proposed GCB-SA and FCB-SA Approaches

Embeddings generated from two pre-trained word embeddings are used to implement proposed DL approaches. Several ensemble neural architectures with Conv1D and Bi-RNN models with SA mechanisms are built and used for experimentation. The description of the datasets used, and experimental setup with various hyper-parameters settings used are mentioned in Section 4.5. The layered schematic for the proposed hybrid DL-based approaches with the SA mechanism for efficient detection of social spam is shown in Figure 4.4.

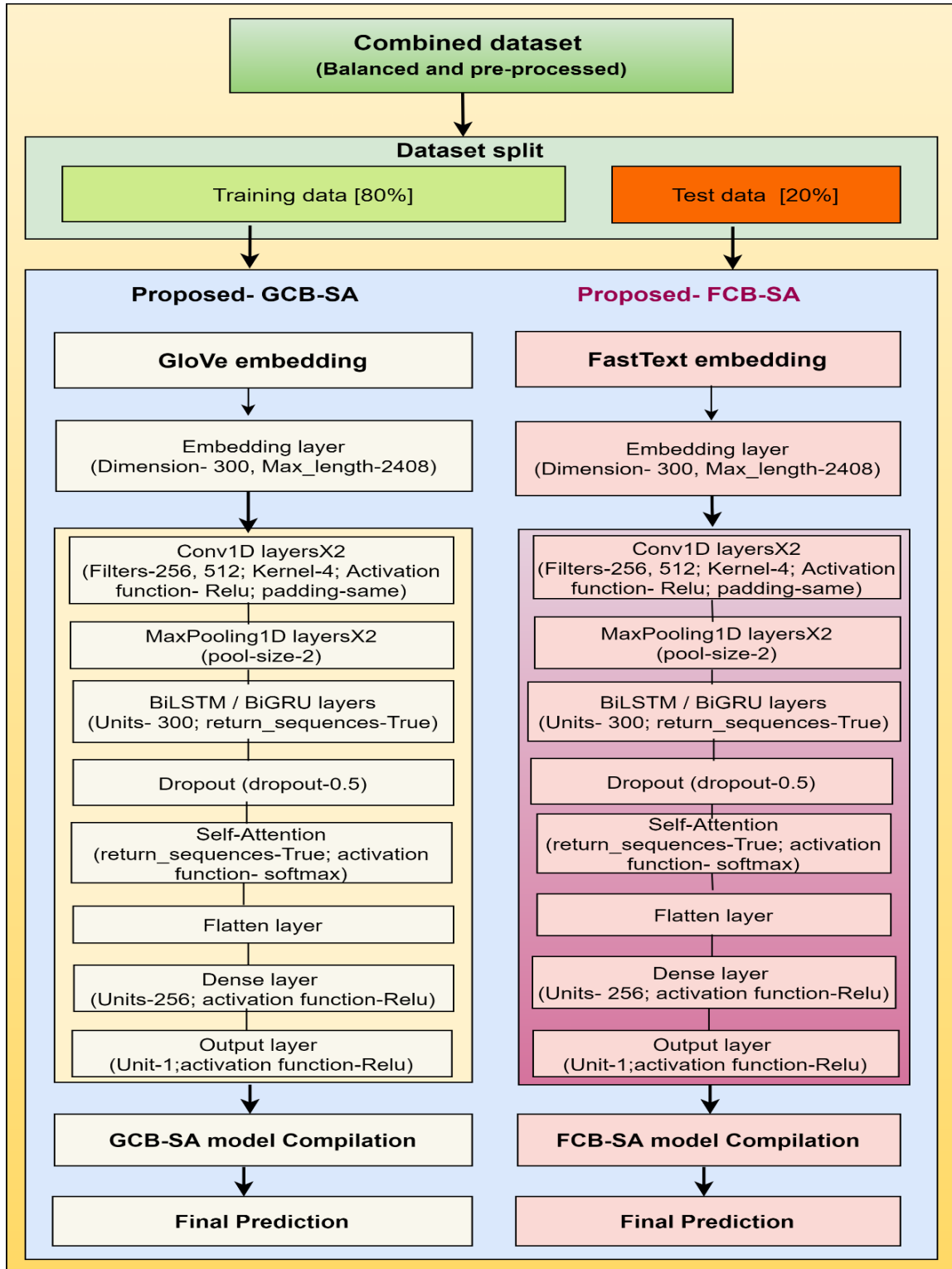


Figure 4.4: Schematic for proposed GCB-SA and FCB-SA approaches

The proposed hybrid DL-based approaches are described as follows:

Proposed GCB-SA: In this hybrid DL-based approach, GloVe embeddings, multiple Conv1D layers, and bi-directional GRU/LSTM layers with the SA mechanism are used. The GloVe embedding of 300-dimension word vectors trained on 6 billion words with 400k dictionary words is used in GCB-SA to generate word vector representations. The parameter values used for the embedding layer are given in Table 4.3. The pre-processed data is initially passed to the GloVe embeddings to generate an embedding matrix on a balanced *combined dataset*. The embeddings are then passed as input into the first layer of the sequential model. The sequential model comprises several Conv1D and max pooling layers that work together to make a feature map over the sentence. The feature map explicitly captures short-range and long-range relationships between words or phrases. The captured feature maps are passed to Bi-GRU and Bi-LSTM layers. Eventually, the SA mechanism is applied to take inputs from Bi-GRU and Bi-LSTM layers to extract the most significant inputs and aggregate their representations using attention weights and bias to form output as a sentence vector. The *softmax()* function normalizes the results into probability distribution on all self-attention scores and is multiplied by the value vector. The weighted value vectors are then summed up to get the SA output for the word. The same process is applied to all input sequences. Finally, flatten and dense layers are added with activation functions to get the final predictions. The optimized hyper-parameters and model compilation parameters used for the GCB-SA approach are given in Table 4.3. The summary of the proposed GCB-SA approach used in the framework for social spam detection is given in Table 4.1.

Proposed FCB-SA: In this hybrid DL-based approach, FastText embeddings, multiple Conv1D layers, and bi-directional GRU/LSTM layers with a SA mechanism are used. The FastText embedding is huge but comparatively fast to train than other pre-trained word embeddings. FastText embedding with 300-dimensional word vectors is used to generate word vectors. Besides, FastText can generate vector representations for words not included in the model dictionary, using sub-words (n-grams) instead of individual words. Compared

with other embedding techniques, FastText helps build vectors for unknown words since the n-gram exceeds the number of words. The parameter values used for the FastText embedding layer are given in Table 4.3. The embedding layer generated from FastText is considered as an input to the first layer of the sequential model. The layered architecture of the sequential model used in FCB-SA is the same as in the GCB-SA approach. This approach also uses the SA mechanism to extract the most significant inputs. The representations based on attention weights and bias are aggregated to form an output sentence vector. The results are normalized into a probability distribution using the *softmax()* function to all the self-attention scores and multiplying the results by the value vector. The SA output for the word is obtained by summing the weighted value vectors. All input sequences undergo the same procedure. Finally, flatten and dense layers are added with activation functions to get the final predictions. The optimized hyper-parameters and model compilation parameters used for the FCB-SA approach are given in Table 4.3. The summary of the proposed FCB-SA approach used in the framework for social spam detection is given in Table 4.2.

Table 4.1: Summary of the proposed GCB-SA approach

Model: GCB-SA		
Layer (type)	Output Shape	Param #
embedding_5 (Embedding)	(None, 2408, 300)	15582000
conv1d_10 (Conv1D)	(None, 2408, 512)	614912
max_pooling1d_10 (MaxPooling1D)	(None, 1204, 512)	0
conv1d_11 (Conv1D)	(None, 1204, 256)	524544
max_pooling1d_10 (MaxPooling1D)	(None, 602, 256)	0
bidirectional_10 (Bidirectional)	(None, 602, 600)	1336800
bidirectional_11 (Bidirectional)	(None, 602, 600)	1623600
dropout (Dropout)	(None, 602, 600)	0
attention_5 (Attention)	(None, 602, 600)	1202
flatten_5 (Flatten)	(None, 361200)	0
dense_10 (Dense)	(None, 256)	92467456
dense_11 (Dense)	(None, 1)	257

Total params: 112,150,771
Trainable params: 96,568,771
Non-trainable params: 15,582,000

Table 4.2: Summary of the proposed FCB-SA approach

Model: FCB-SA		
Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 2408, 300)	17931600
conv1d (Conv1D)	(None, 2408, 512)	614912
max_pooling1d (MaxPooling1D)	(None, 1204, 512)	0
conv1d (Conv1D)	(None, 1204, 256)	524544
max_pooling1d (MaxPooling1D)	(None, 602, 256)	0
bidirectional (Bidirectional)	(None, 602, 600)	1336800
bidirectional (Bidirectional)	(None, 602, 600)	1623600
dropout (Dropout)	(None, 602, 600)	0
attention (Attention)	(None, 602, 600)	1202
flatten (Flatten)	(None, 361200)	0
dense (Dense)	(None, 256)	92467456
dense (Dense)	(None, 1)	257
Total params: 114,500,371		
Trainable params: 96,568,771		
Non-trainable params: 17,931,600		

4.5 Experiment Setup and Results

4.5.1 Dataset description

Due to a lack of a bigger social spam corpus, the *combined dataset* is merged from SMS Spam, Ling Spam, and Fake news datasets. Thus balanced *combined dataset* retrieved using the ST technique is used for DL experimentation. The balanced *combined dataset* is divided into training data (80%) and testing data (20%).

4.5.2 Experimental Settings

The individual DL models and proposed hybrid approaches are performed with Python 3.7.

The GPU used for experimentation is Tesla T4. Experiments are repeated with varied hyperparameter values until optimal performance is reached. The optimal hyperparameter settings for the proposed hybrid DL approaches are given in Table 4.3.

Table 4.3: Optimised hyperparameter settings used for building and testing proposed hybrid DL-based approaches

Parameter	Value	Layer
Embedding dimensions	300	GloVe and FastText embedding
vocab_length	51940	GloVe embedding
vocab_length	59772	FastText embedding
max_length	2408	GloVe and FastText embedding
Trainable	False	GloVe and FastText embedding
Filters	256	Conv1D
Filters	512	Conv1D
Kernel_size	4	Conv1D
Activation function	Relu	Conv1D
Padding	Same	Conv1D
MaxPooling1D (pool_size)	2	MaxPooling x 2
Bi-GRU units	300	Bi-GRU
Bi-LSTM units	300	Bi-LSTM
return_sequences	True	Bi-LSTM/Bi-GRU
Dropout	0.5	Dropout
return_sequences	True	Self-attention
Activation function	<i>Softmax()</i>	Self-attention
Dense units	256	Dense
Activation function	Relu	Dense
Dense units	1	Output layer
Output Layer activation function	Sigmoid	Output layer
Learning rate	0.001	Compile
Optimizer	Adam	Compile
Loss	Binary_cross_entropy	Compile
Batch size	128	Fit
Epochs	20	Fit
Validation split	0.1	Fit

4.5.3 Results and Discussions

DL-based experimentation is performed on a balanced *combined dataset* since ML models perform well on this dataset to improve social spam detection performance. Table 4.4 compares different DL baselines and proposed hybrid DL approaches (GCB-SA and FCB-SA) on the *combined dataset* with the top five best performance scores highlighted.

Table 4.4: Performance comparison of various DL baseline approaches with proposed hybrid DL-based approaches

Performance parameters→ Hybrid DL approaches ↓	A (%)	P (%)	R (%)	F (%)	ROC-AUC Score	Training Loss	Validation Loss
GloVe+RNN	76.85	73.48	52.23	67.75	0.78	0.4123	0.4977
GloVe+LSTM	69.19	67.22	68.10	55.32	0.71	0.6019	0.6176
GloVe+GRU	69.73	68.64	69.43	54.98	0.72	0.6056	0.6175
GloVe+Bi-LSTM	94.30	91.23	93.17	91.46	0.97	0.1430	0.1980
GloVe+Bi-GRU	94.86	92.80	94.80	93.62	0.97	0.1221	0.2012
GloVe+CNN	92.74	90.64	91.64	90.52	0.95	0.0825	0.3344
GCB-SA (Proposed)	96.19	94.28	95.36	96.10	0.99	0.1675	0.2801
FastText+RNN	74.22	73.15	76.24	66.35	0.77	0.5984	0.6179
FastText+LSTM	69.54	70.38	72.45	64.44	0.72	0.6214	0.6181
FastText+GRU	70.47	71.56	72.69	68.53	0.73	0.6211	0.6172
FastText+Bi-LSTM	94.56	91.70	93.81	92.70	0.97	0.1512	0.1946
FastText+Bi-GRU	95.24	93.32	94.23	94.28	0.98	0.1363	0.2621
FastText+CNN	93.19	88.21	89.45	89.24	0.96	0.2763	0.2652
FCB-SA (Proposed)	97.26	96.86	95.62	98.60	0.99	0.0836	0.2327

Table 4.4 shows that the performance of the proposed FCB-SA approach outperforms with an accuracy of 97.26% on the balanced *combined dataset* compared to other DL-based approaches. Some other good-performing DL approaches are GCB-SA, FastText+Bi-

GRU, GloVe+Bi-GRU, and FastText+Bi-LSTM. However, the performance gradually increased when pre-trained word embeddings from GloVe and FastText are passed in the hybrid DL approaches such as Bi-LSTM, Bi-GRU, and CNN. The Bi-GRU approach yields better accuracy on both GloVe and FastText word embeddings. Few DL models perform better with FastText than GloVe word embeddings. The training and validation loss of baseline DL approaches and proposed approaches are also mentioned in Table 4.4. The difference between training and validation loss of DL approaches is less, which shows a good fit. The results are further improved with the self-attention mechanism as the emphasis is given to significant words or input vectors. Hence, the proposed DL approaches, i.e., GCB-SA and FCB-SA, are combined with the Conv1D and attention layers. After several iterations, the best optimal hyperparameters for the proposed GCB-SA and FCB-SA approaches are gathered and presented in Table 4.3. The consolidated accuracy analysis for baseline DL approaches and proposed hybrid DL approaches (GCB-SA and FCB-SA) are shown in Figure 4.7.

The graphical comparison between training accuracy and validation accuracy for proposed DL-based hybrid approaches, i.e., GCB-SA and FCB-SA on the balanced *combined dataset*, is presented in Figure 4.5. From Figure 4.5, we found that the training accuracies of both proposed approaches are nearly the same, whereas the validation accuracy of FCB-SA progressively improves with more epochs.

The training and validation loss observed on 20 epochs for the proposed DL-based hybrid approaches, i.e., GCB-SA and FCB-SA on the balanced *combined dataset*, is presented in Figure 4.6. From Figure 4.6, we observe that training loss in both proposed approaches is almost the same, whereas validation loss is slightly less in GCB-SA than in FCB-SA with increasing epochs.

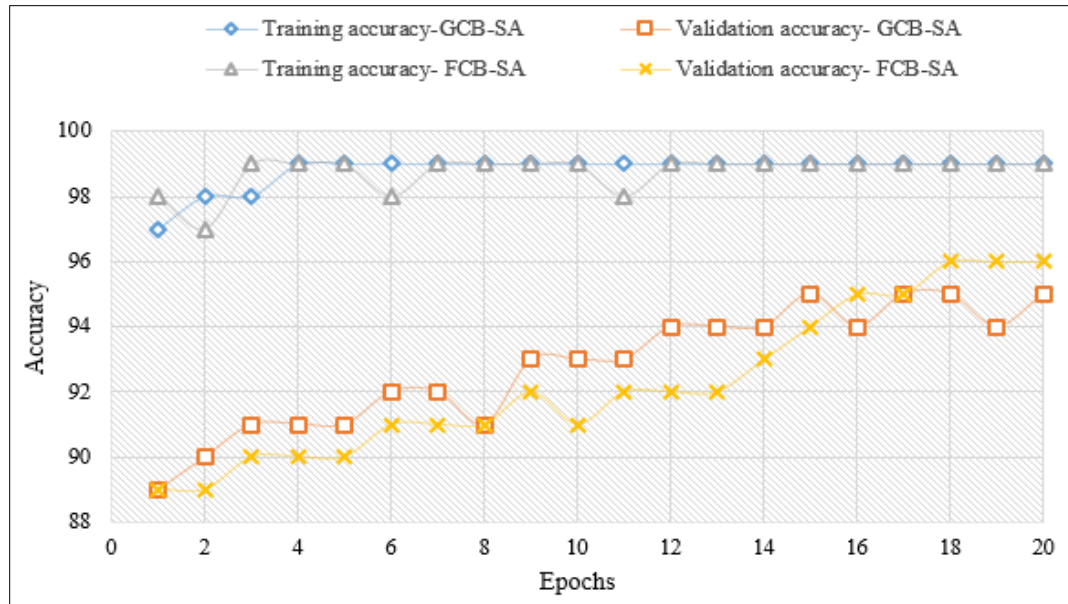


Figure 4.5: Comparison of training accuracy and validation accuracy for proposed hybrid DL-based approaches

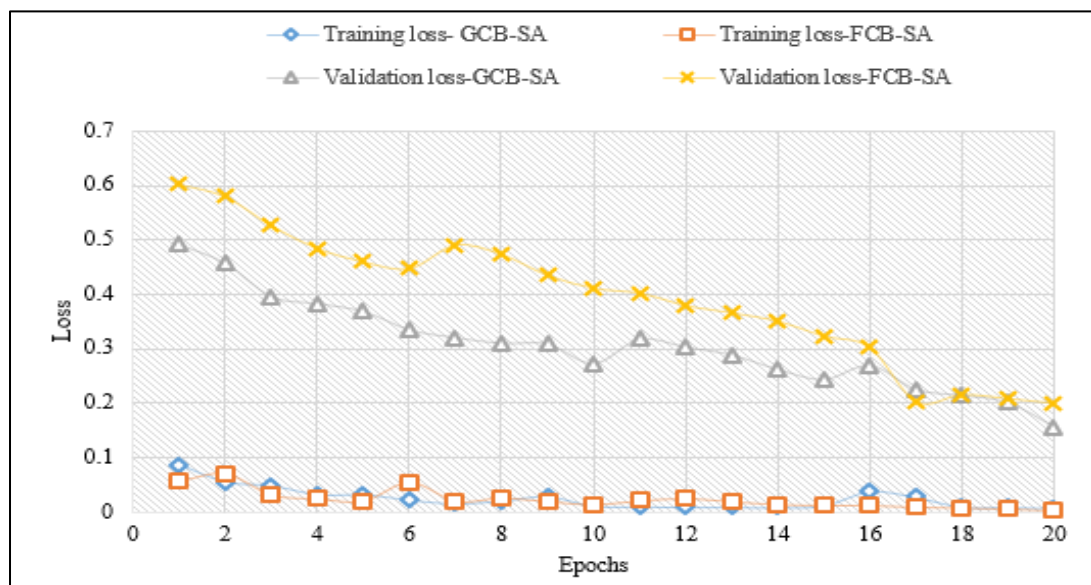


Figure 4.6: Comparison of training loss and validation loss for proposed hybrid DL-based approaches

The consolidated accuracy analysis is carried out by graphically comparing GCB-SA and FCB-SA with other good-performing DL approaches on the balanced *combined dataset*, as shown in Figure 4.7. The DL models with 90% or above accuracy are considered good and compared with the proposed GCB-SA and FCB-SA. The performance of GCB-SA and FCB-SA hybrid DL approaches are highlighted with data labels to have a quick comparison with other DL approaches on the balanced *combined dataset*. Furthermore, the analysis reveals that the hybrid DL approaches such as Bi-LSTM, Bi-GRU, and CNN produced considerably good results with pre-trained word embeddings from GloVe and FastText.

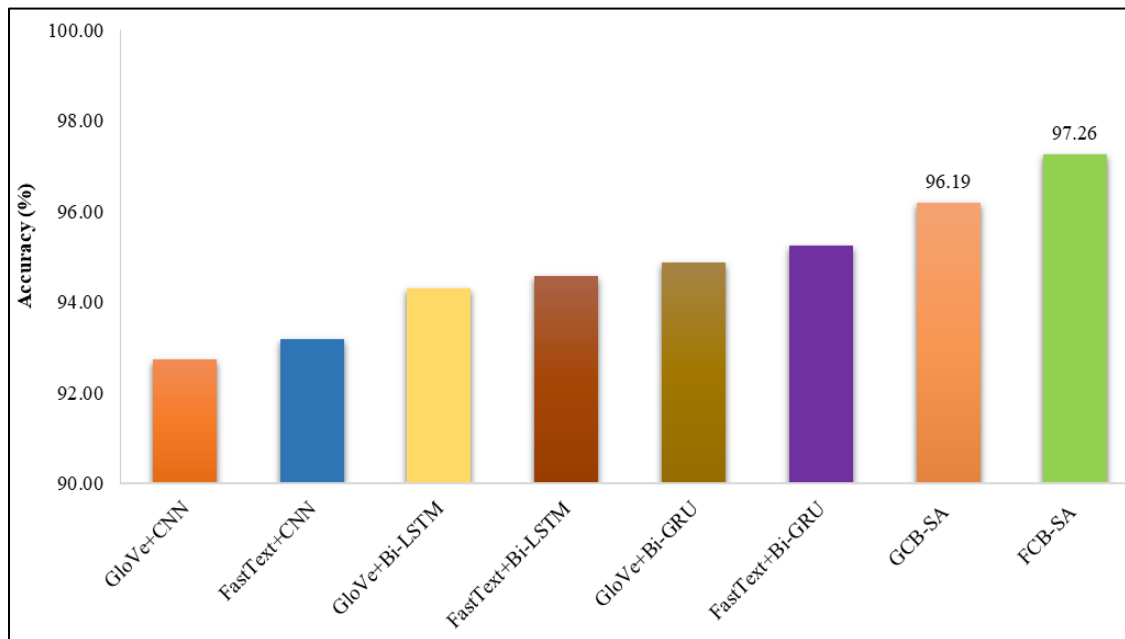


Figure 4.7: Consolidated accuracy analysis of proposed hybrid DL-based approaches with DL baseline approaches

4.5.4 Comparison with existing studies

The performance comparison of proposed hybrid DL-based approaches with some existing approaches is given in Table 4.5.

Table 4.5: Performance comparison of proposed hybrid DL-based approaches with existing approaches

References	Feature extraction/ Optimization technique	ML/DL Classifier	Dataset/ Balancing technique	A (%)
(Lai et al., 2022)	TF-IDF and Word2Vec	LR,NB,SVM, RF, CNN, LSTM	<i>Fake news dataset</i>	90.00
[147]	Word2Vec, WordNet and ConceptNet	SCNN Framework, KNN,	<i>Twitter Dataset</i>	94.40
		NB, ANN, RF, SVM	<i>SMS spam dataset</i>	86.5
[246]	TF-IDF	MOEFS+RDNN	<i>Hyves dataset</i>	90.02
			<i>Twitter dataset</i>	95.60
GCB-SA (Proposed)	GloVe	Conv1D, Bi-GRU, Bi- LSTM, SA mechanism	<i>Combined dataset with ST</i>	96.19
FCB-SA (Proposed)	FastText	Conv1D, Bi-GRU, Bi- LSTM, SA mechanism	<i>Combined dataset with ST</i>	97.26

4.6 Summary

The proposed ML-based voting models PHVE and PSVE provided better results than baseline ML models over balanced datasets retrieved using the ST data sampling technique. However, DL-based experimentation is employed on the combined dataset retrieved using the ST technique to further improve the performance of social spam detection. The DL model takes input data and extracts relevant features for faster learning. Therefore, GloVe and FastText pre-trained word embeddings with individual DL models such as RNN, GRU, LSTM, Bi-GRU, Bi-LSTM, and CNN are experimented on the balanced *combined dataset*. Also, two hybrid DL approaches, GCB-SA and FCB-SA, using GloVe and FastText word embeddings and a sequential ensemble model with the self-attention mechanism, are proposed. The sequential model includes multiple Conv1D, Bi-GRU/Bi-LSTM model layers with the self-attention mechanism. The Conv1D layer slides different kernels across

a sequence, generating 1D feature map kernel. Each kernel learned to detect a short sequential pattern. The short input sequences from Conv1D further helped Bi-GRU and Bi-LSTM layers to detect longer patterns. The Bi-GRU/LSTM have comparatively long-term memory than simple RNNs and can work in both forward and backward directions to provide better performance. One dropout layer is added to the sequential model to avoid overfitting. The proposed hybrid approaches are built by adding layers from CNN, Bi-GRU, and Bi-LSTM with optimized hyperparameters.

DL models with FastText embeddings perform better than DL models with GloVe, with a variation of approximately (1 ~ 2 %). The SA mechanism further improves the results as the emphasis is given to significant words or input vectors. The attention mechanism improved performance dynamically by emphasizing a few tokens in input sequences using transforming token embeddings (Bahdanau et al., 2014). The proposed FCB-SA outperformed other DL approaches with an accuracy of 97.26 % on a balanced *combined dataset* retrieved using the ST technique. The proposed FCB-SA has a validation loss of 0.2327, which is acceptable. Experimentation results show that our FCB-SA outperforms other existing frameworks/approaches.

Chapter 5 : Conclusions and Future Scope

This thesis focuses on efficiently identifying the text spam content in OSNs. This chapter concludes the research work presented in this thesis and outlines the future scope of the presented work.

5.1 Conclusions

This thesis introduced background related to social spam and spamming process. A social spam taxonomy was proposed, highlighting different accounts and user engagements to instigate spam and associated cyber-attacks. The important challenges and open issues related to spam detection in OSN were thoroughly discussed. A comprehensive review of ML/DL-based techniques for social spam and spammer detection approaches covered advanced AI-enabled Deepfakes and social spambots. A coherent analysis for social spam detection included information related to existing benchmark datasets, features employed, ML/DL models, performance measures, merits, and demerits. The Literature review revealed that SL techniques had been used extensively. However, SL techniques are sometimes ineffective due to their intrinsic challenges comprising real-time data gathering, a scarcity of ground truth large datasets, a limited feature set, the class imbalance problem, performance issues, adversarial attacks on ML models, scaling concerns, model robustness, etc. USL techniques were used to find groups of spammers or bot accounts but required huge data storage. SSL techniques can be a better option in the absence of publicly labeled data and have been used to bypass the limitations of SL and USL. The DL models are recommended over ML models due to better feature extraction and faster spam detection over OSNs. However, DL models usually require more data than standard ML models. For better training and testing of ML/DL models, large-scale ground truth data is needed. It's observed that most of the ML/DL-based approaches for social spam detection were trained and evaluated on a small dataset without considering the class imbalance issue. These major challenges and issues have been deliberated and critically discussed.

This thesis proposed a framework with ML and DL approaches to categorize text message as *Spam* and *Ham*. For ML-based approaches, two voting ensemble models (PHVE and PSVE) were proposed over the larger and more diverse dataset. The new dataset was built by merging the three benchmark datasets to build large *spam* and *ham*-labeled corpus *combined dataset*. The class imbalance problem was solved using NearMiss and SmoteTomek data resampling techniques. The best ML models from varied approaches were selected, trained, and tested individually on imbalanced and balanced datasets. Later, the proposed voting ensemble models combined the performance of multiple models to get the most accurate prediction results. The comparative results of different ML models with PHVE and PSVE on various imbalanced and balanced datasets have been presented and discussed. The performance of proposed voting ensemble models and other baselines such as PHVE, ETC, SVM, and RF considerably improved on balanced *combined dataset* retrieved using the ST technique, except KNN. The KNN model performed better on the imbalanced *Ling spam dataset* with an accuracy of 90.70% than the balanced *Ling spam datasets*. PSVE model outperformed other ML baseline models with an accuracy of 95.88% on the balanced *combined dataset*. The ML models produced accurate and useful results but could not learn low-level features, which is not a problem for DL models.

DL-based experimentation was employed on the *combined dataset* retrieved using the ST technique to further improve social spam detection performance. GloVe and FastText, pre-trained word embeddings, were generated on the balanced *combined dataset* and passed as embedding in various DL models such as RNN, GRU, LSTM, Bi-GRU, Bi-LSTM, and CNN. Also, two hybrid DL-based approaches, GCB-SA and FCB-SA, were proposed using GloVe and FastText word embeddings. The proposed hybrid DL approach included a sequential model with multiple Conv1D, Bi-GRU/Bi-LSTM model layers with optimized hyperparameters and a self-attention mechanism. The Conv1D generated 1D feature map kernels by sliding kernels across a sequence. Each kernel detected short sequential patterns. Conv1D's short input sequences helped Bi-GRU and Bi-LSTM layers find longer patterns. Bi-GRU/LSTMs have longer memory than RNNs and can work in both ways to improve the efficiency. The experimentation revealed that DL models with FastText embeddings

performed better than DL models with GloVe, with a variation of approximately (1 ~ 2 %). The results were further improved using the SA mechanism, as the emphasis was given to significant words or input vectors. The proposed FCB-SA outperforms other DL approaches with an accuracy of 97.26 % and a validation loss of 0.2327. The proposed DL-based approaches performed better than the existing and proposed ML models. Extensive ML and DL experimentation and performance comparisons concluded that the proposed framework efficiently improved the performance of social spam detection by addressing some prominent issues. However, work can still be advanced by incorporating the future directions outlined in the subsequent section.

5.2 Future Scope

Acquainted with our proposed approaches, we suggest the following directions to advance research in social spam detection:

- The word embeddings, such as GloVe and FastText, provided good results. However, it can be observed how proposed approaches performed with other contextual pre-trained word embeddings in the future.
- The proposed approaches are evaluated on binary datasets. However, the work can be extended to multiclass, multilingual, and multimodal (text, image, audio, and video) datasets.
- Advanced NLP-based transformer models can also be examined.
- Research can be advanced on Deepfake/forged video detection using various techniques/approaches reconnoitered in the Literature review section.
- OSN user behavior can be analyzed to detect cyber abuse, such as hate speech, sarcasm, and harassment.
- The work can be advanced by developing a browser extension, web application, or functionality in the existing OSN applications that will automatically identify spam or fake content based on its probability of being spam/fake.
- Blockchain-based systems can also be investigated due to promising security results.

References

- [1] F. Persia and D. D'Auria, "A survey of online social networks: Challenges and opportunities," *Proc. - 2017 IEEE Int. Conf. Inf. Reuse Integr. IRI 2017*, vol. 2017-Janua, no. Figure 1, pp. 614–620, 2017.
- [2] "Biggest social media platforms 2022 | Statista," *Statista*, 2022, 2022. [Online]. Available: <https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/>. [Accessed: 14-Oct-2022].
- [3] R. Gallotti, F. Valle, N. Castaldo, P. Sacco, and M. De Domenico, "Assessing the risks of 'infodemics' in response to COVID-19 epidemics," *Nat. Hum. Behav.*, 2020.
- [4] M. Westerlund, "The emergence of deepfake technology: A review," *Technol. Innov. Manag. Rev.*, vol. 9, no. 11, pp. 39–52, 2019.
- [5] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 3204–3213, Sep. 2019.
- [6] S. Rao, A. K. Verma, and T. Bhatia, "Online social networks misuse, cyber crimes, and counter mechanisms," in *Analyzing Global Social Media Consumption*, vol. i, IGI Global, 2020, pp. 183–203.
- [7] S. Rao, A. K. Verma, and T. Bhatia, "Evolving Cyber Threats, Combating Techniques, and Open Issues in Online Social Networks," in *Handbook of Research on Cyber Crime and Information Privacy*, IGI Global, 2020, pp. 219–235.
- [8] C. Watson and T. Zaw, *OWASP Automated Threat Handbook*. 2018.

- [9] MITRE Corporation, “Matrix - Enterprise | MITRE ATT&CK®,” 2019. [Online]. Available: <https://attack.mitre.org/matrices/enterprise/>. [Accessed: 08-Jul-2020].
- [10] Twitter, “Platform manipulation and spam policy,” 2019. [Online]. Available: <https://help.twitter.com/en/rules-and-policies/platform-manipulation>. [Accessed: 15-Aug-2020].
- [11] Facebook, “Spam | Facebook Help Centre | Facebook,” 2020. [Online]. Available: <https://www.facebook.com/help/287137088110949>. [Accessed: 15-Aug-2020].
- [12] NexGate, “Research Report 2013 State of Social Media Spam,” vol. 33, no. 3, pp. i–iii, 2013.
- [13] P. Wagner, “Social Media Users Notice Spam Increase | Statista,” 2018. [Online]. Available: <https://www.statista.com/chart/15088/level-of-spam-perception-in-us-uk-ca/>. [Accessed: 09-Oct-2020].
- [14] M. Armstrong, “16% of All Facebook Accounts Are Fake or Duplicates,” *Statista*, 2020. [Online]. Available: <https://www.statista.com/chart/20685/duplicate-and-false-facebook-accounts/>. [Accessed: 17-Jan-2021].
- [15] Spamhaus, “The Spamhaus Project - The Top 10 Most Abused TLDs,” 2020. [Online]. Available: <https://www.spamhaus.org/statistics/tlds/>. [Accessed: 24-Oct-2020].
- [16] Trend Micro, “Youtube, Twitter Hunt Down Deepfakes - Security News - Trend Micro USA,” 2020. [Online]. Available: <https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/youtube-twitter-hunt-down-deepfakes>. [Accessed: 16-Aug-2020].
- [17] S. Frenkel, D. Alba, and R. Zhong, “Surge of Virus Misinformation Stumps Facebook and Twitter,” *The New York Times*, 2020. [Online]. Available: <https://www.nytimes.com/2020/03/08/technology/coronavirus-misinformation->

social-media.html. [Accessed: 16-Aug-2020].

- [18] Facebook, “August 2020 Coordinated Inauthentic Behavior Report - About Facebook,” 2020.
- [19] F. Richter, “Chart: Toxic Content Runs Rampant on Facebook | Statista,” *Statista*, 2020. [Online]. Available: <https://www.statista.com/chart/13875/facebooks-spring-cleaning/>. [Accessed: 17-Jan-2021].
- [20] L. F. Cranor and B. A. Lamacchia, “Spam!,” *Commun. ACM*, vol. 41, no. 8, pp. 74–83, Aug. 1998.
- [21] N. Spirin and J. Han, “Survey on web spam detection,” *ACM SIGKDD Explor. Newsl.*, vol. 13, no. 2, pp. 50–64, 2012.
- [22] M. Chakraborty, S. Pal, R. Pramanik, and C. Ravindranath Chowdary, “Recent developments in social spam detection and combating techniques: A survey,” *Inf. Process. Manag.*, vol. 52, no. 6, pp. 1053–1073, Nov. 2016.
- [23] P. Hayati, V. Potdar, A. Talevski, N. Firoozeh, S. Sarenche, and E. A. Yeganeh, “Definition of Spam 2.0: New spamming boom,” in *4th IEEE International Conference on Digital Ecosystems and Technologies - Conference Proceedings of IEEE-DEST 2010, DEST 2010*, 2010, pp. 580–584.
- [24] P. Hayati and V. Potdar, “Spam 2.0 state of the art,” in *Emerging Digital Forensics Applications for Crime Detection, Prevention, and Security*, IGI Global, 2012, pp. 103–121.
- [25] E. Ferrara, “The history of digital spam,” *Communications of the ACM*, vol. 62, no. 8. Association for Computing Machinery, pp. 82–91, 01-Aug-2019.
- [26] D. Ramalingam and V. Chinnaiah, “Fake profile detection techniques in large-scale online social networks: A comprehensive review,” *Comput. Electr. Eng.*, vol.

65, no. 3, pp. 165–177, 2018.

- [27] J. Cao, Q. Fu, Q. Li, and D. Guo, “Discovering hidden suspicious accounts in online social networks,” *Inf. Sci. (Ny)*, vol. 394–395, pp. 123–140, 2017.
- [28] R. Kaur, S. Singh, and H. Kumar, “Rise of spam and compromised accounts in online social networks: A state-of-the-art review of different combating approaches,” *J. Netw. Comput. Appl.*, vol. 112, no. March, pp. 53–88, 2018.
- [29] X. Ruan, Z. Wu, H. Wang, and S. Jajodia, “Profiling Online Social Behaviors for Compromised Account Detection,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 176–187, 2016.
- [30] K. S. Adewole, N. B. Anuar, A. Kamsin, K. D. Varathan, and S. A. Razak, “Malicious accounts: Dark of the social networks,” *J. Netw. Comput. Appl.*, vol. 79, no. September 2016, pp. 41–67, 2017.
- [31] M. Al-Qurishi, M. Al-Rakhami, A. Alamri, M. AlRubaian, S. M. M. Rahman, and M. S. Hossain, “Sybil defense techniques in online social networks: A survey,” *IEEE Access*, vol. 5, pp. 1200–1219, 2017.
- [32] Z. Yamak, J. Saunier, and L. Vercouter, “SocksCatch: Automatic detection and grouping of sockpuppets in social media,” *Knowledge-Based Syst.*, vol. 149, pp. 124–142, Jun. 2018.
- [33] J. Li, W. Zhou, J. Han, and S. Hu, “Sockpuppet detection in social network via propagation tree,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019, vol. 11540 LNCS, pp. 507–513.
- [34] E. Van der Walt, J. H. P. Eloff, and J. Grobler, “Cyber-security: Identity deception detection on social media platforms,” *Comput. Secur.*, vol. 78, pp. 76–89, 2018.

- [35] J. Zhang, R. Zhang, Y. Zhang, and G. Yan, "The rise of social botnets: Attacks and countermeasures," *IEEE Trans. Dependable Secur. Comput.*, vol. 15, no. 6, pp. 1068–1082, Nov. 2018.
- [36] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96–104, 2016.
- [37] S. Cresci, A. Spognardi, M. Petrocchi, M. Tesconi, and R. Di Pietro, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," in *26th International World Wide Web Conference 2017, WWW 2017 Companion*, 2017, pp. 963–972.
- [38] M. Latah, "Detection of malicious social bots: A survey and a refined taxonomy," *Expert Syst. Appl.*, vol. 151, p. 113383, Mar. 2020.
- [39] R. Singh, S. Tanwar, and T. P. Sharma, "Utilization of blockchain for mitigating the distributed denial of service attacks," *Secur. Priv.*, vol. 3, no. 3, p. e96, May 2020.
- [40] S. Barbon, G. F. C. Campos, G. M. Tavares, R. A. Igawa, M. L. Proença, and R. C. Guido, "Detection of human, legitimate bot, and malicious bot in online social networks based on wavelets," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 14, no. 1s, pp. 1–17, 2018.
- [41] Y. Xin, C. Zhao, H. Zhu, and M. Gao, "A Survey of Malicious Accounts Detection in Large-Scale Online Social Networks," *Proc. - 4th IEEE Int. Conf. Big Data Secur. Cloud, BigDataSecurity 2018, 4th IEEE Int. Conf. High Perform. Smart Comput. HPSC 2018 3rd IEEE Int. Conf. Intell. Data Secur.*, pp. 155–158, 2018.
- [42] C. Cao and J. Caverlee, "Detecting spam URLs in social media via behavioral analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9022, pp. 703–714, 2015.

- [43] S. H. Qi, L. AlKulaib, and D. A. Broniatowski, “Detecting and characterizing bot-like behavior on twitter,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10899 LNCS, pp. 228–232, 2018.
- [44] M. Kantepe and M. C. Gañiz, “Preprocessing framework for Twitter bot detection,” *2nd Int. Conf. Comput. Sci. Eng. UBMK 2017*, pp. 630–634, 2017.
- [45] A. Bessi and E. Ferrara, “Social bots distort the 2016 U.S. Presidential election online discussion,” *First Monday*, vol. 21, no. 11, 2016.
- [46] K. C. Yang, P. M. Hui, and F. Menczer, “Bot electioneering volume: Visualizing social bot activity during elections,” in *The Web Conference 2019 - Companion of the World Wide Web Conference, WWW 2019*, 2019.
- [47] M. Bakardjieva, “Rationalizing Sociality: An Unfinished Script for Socialbots,” *Inf. Soc.*, vol. 31, no. 3, pp. 244–256, May 2015.
- [48] Freelancers, “Hire Freelancers & Find Freelance Jobs Online | Freelancer,” 2020. [Online]. Available: <https://www.freelancer.com/>. [Accessed: 22-Aug-2020].
- [49] Sandaha, “<http://www.sandaha.cc/>,” 2020. [Online]. Available: <http://www.sandaha.cc/>. [Accessed: 22-Aug-2020].
- [50] Amazon, “Amazon Mechanical Turk,” 2020. [Online]. Available: <https://www.mturk.com/>. [Accessed: 22-Aug-2020].
- [51] B. Liu *et al.*, “Co-Detection of crowdturfing microblogs and spammers in online social networks,” *World Wide Web*, vol. 23, no. 1, pp. 573–607, 2020.
- [52] R. P. Khandpur, T. Ji, S. Jan, G. Wang, C. T. Lu, and N. Ramakrishnan, “Crowdsourcing cybersecurity: Cyber attack detection using social media,” in *International Conference on Information and Knowledge Management, Proceedings*, 2017, vol. Part F1318, pp. 1049–1057.

- [53] M. Singh, D. Bansal, and S. Sofat, “Followers or Fradulents? An Analysis and Classification of Twitter Followers Market Merchants,” *Cybern. Syst.*, vol. 47, no. 8, pp. 674–689, 2016.
- [54] F. Salahdine and N. Kaabouch, “Social engineering attacks: A survey,” *Future Internet*, vol. 11, no. 4. MDPI AG, p. 89, 02-Apr-2019.
- [55] N. Mashtalyar, U. N. Ntaganzwa, T. Santos, S. Hakak, and S. Ray, “Social Engineering Attacks: Recent Advances and Challenges,” pp. 417–431, Jul. 2021.
- [56] G. S. Thejas, K. G. Boroojeni, K. Chandna, I. Bhatia, S. S. Iyengar, and N. R. Sunitha, “Deep learning-based model to fight against Ad click fraud,” *ACMSE 2019 - Proc. 2019 ACM Southeast Conf.*, no. April, pp. 176–181, 2019.
- [57] C. Crane, “The Definitive Cyber Security Statistics Guide for 2020 - Security Boulevard,” 2020. [Online]. Available: <https://securityboulevard.com/2020/05/the-definitive-cyber-security-statistics-guide-for-2020/>. [Accessed: 12-Jul-2020].
- [58] A. Ioannou, J. Blackburn, G. Stringhini, E. De Cristofaro, N. Kourtellis, and M. Sirivianos, “From risk factors to detection and intervention: a practical proposal for future work on cyberbullying,” *Behav. Inf. Technol.*, vol. 37, no. 3, pp. 258–266, Mar. 2018.
- [59] M. Fire, R. Goldschmidt, and Y. Elovici, “Online social networks: Threats and solutions,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 4, pp. 2019–2036, 2014.
- [60] R. Prasad and V. Rohokale, “Cyber Threats and Attack Overview,” in *Cyber Security: The Lifeline of Information and Communication Technology. Springer Series in Wireless Technology.*, Springer, Cham, 2020, pp. 15–31.
- [61] C. V. Bijitha, R. Sukumaran, and H. V. Nath, “A survey on ransomware detection techniques,” in *Communications in Computer and Information Science*, 2020, vol. 1186 CCIS, pp. 55–68.

- [62] Y. Yigit, B. Bal, A. Karameseoglu, T. Q. Duong, and B. Canberk, “Digital Twin-Enabled Intelligent DDoS Detection Mechanism for Autonomous Core Networks,” *IEEE Commun. Stand. Mag.*, vol. 6, no. 3, pp. 38–44, Sep. 2022.
- [63] I. Kayes and A. Iamnitchi, “Privacy and security in online social networks: A survey,” *Online Soc. Networks Media*, vol. 3–4, no. 4, pp. 1–21, 2017.
- [64] M. C. Libicki, “Drawing inferences from cyber espionage,” in *International Conference on Cyber Conflict, CYCON*, 2018, vol. 2018-May, pp. 109–121.
- [65] D. Herrmann, “Cyber Espionage and Cyber Defence,” in *Information Technology for Peace and Security*, Springer Fachmedien Wiesbaden, 2019, pp. 83–106.
- [66] D. Pacheco, P.-M. Hui, C. Torres-Lugo, B. T. Truong, A. Flammini, and F. Menczer, “Uncovering Coordinated Networks on Social Media: Methods and Case Studies,” *Proc. Int. AAAI Conf. Web Soc. Media*, vol. 15, pp. 455–466, Jan. 2021.
- [67] E. Ferrara, “Disinformation and social bot operations in the run up to the 2017 French presidential election,” *First Monday*, vol. 22, no. 8, Aug. 2017.
- [68] N. Z. Gong and B. Liu, *You are who you know and how you behave: Attribute inference attacks via users’ social friends and behaviors*. 2016.
- [69] N. Z. Gong and B. Liu, “Attribute inference attacks in online social networks,” *ACM Trans. Priv. Secur.*, vol. 21, no. 1, 2018.
- [70] M. Fire and R. Puzis, “Organization Mining Using Online Social Networks,” *Networks Spat. Econ.*, vol. 16, no. 2, pp. 545–578, 2016.
- [71] R. Sánchez-Corcuera, A. Bilbao-Jayo, U. Zulaika, and A. Almeida, “Analysing centralities for organisational role inference in online social networks,” *Eng. Appl. Artif. Intell.*, vol. 99, no. July 2020, p. 104129, 2021.
- [72] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O.

- E. Ajibuwa, “Machine learning for email spam filtering: review, approaches and open research problems,” *Heliyon*, vol. 5, no. 6, 2019.
- [73] A. Fronzetti Colladon and P. A. Gloor, “Measuring the impact of spammers on e-mail and Twitter networks,” *Int. J. Inf. Manage.*, no. April 2017, pp. 0–1, 2018.
- [74] C. Grimme, M. Preuss, L. Adam, and H. Trautmann, “Social Bots: Human-Like by Means of Human Control?,” *Big Data*, vol. 5, no. 4, pp. 279–293, Dec. 2017.
- [75] S. Bazzaz Abkenar, M. Haghi Kashani, E. Mahdipour, and S. M. Jameii, “Big data analytics meets social media: A systematic review of techniques, open issues, and future directions,” *Telemat. Informatics*, vol. 57, no. June, p. 101517, 2021.
- [76] G. Lin, N. Sun, S. Nepal, J. Zhang, Y. Xiang, and H. Hassan, “Statistical Twitter Spam Detection Demystified: Performance, Stability and Scalability,” *IEEE Access*, vol. 5, 2017.
- [77] K. Agrawal *et al.*, “A Comparison of Class Imbalance Techniques for Real-World Landslide Predictions,” *Proc. - 2017 Int. Conf. Mach. Learn. Data Sci. MLDS 2017*, vol. 2018-January, pp. 1–8, Mar. 2018.
- [78] K. Agarwal, P. Uniyal, S. Virendrasingh, S. Krishna, and V. Dutt, “Spam Mail Classification Using Ensemble and Non-Ensemble Machine Learning Algorithms,” *Lect. Notes Networks Syst.*, vol. 141, pp. 179–189, 2021.
- [79] N. H. Imam and V. G. Vassilakis, “A survey of attacks against twitter spam detectors in an adversarial environment,” *Robotics*, vol. 8, no. 3, 2019.
- [80] P. Dasgupta and J. B. Collins, “A survey of game theoretic approaches for adversarial machine learning in cybersecurity tasks,” *AI Magazine*, vol. 40, no. 2, AI Access Foundation, pp. 31–43, 05-Jul-2019.
- [81] T. Miyato, A. M. Dai, and I. Goodfellow, “Adversarial training methods for semi-

- supervised text classification,” *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, pp. 1–11, 2017.
- [82] A. Abusnaina, A. Khormali, H. Alasmay, J. Park, A. Anwar, and A. Mohaisen, “Adversarial learning attacks on graph-based IoT malware detection systems,” in *Proceedings - International Conference on Distributed Computing Systems*, 2019, vol. 2019-July, pp. 1296–1305.
 - [83] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, “On the effectiveness of machine and deep learning for cyber security,” *Int. Conf. Cyber Conflict, CYCON*, vol. 2018-May, pp. 371–389, 2018.
 - [84] M. Mazza, S. Cresci, M. Avvenuti, W. Quattrociocchi, and M. Tesconi, “RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter,” no. February, 2019.
 - [85] R. Sahay, R. Mahfuz, and A. El Gamal, “A Computationally Efficient Method for Defending Adversarial Deep Learning Attacks,” Jun. 2019.
 - [86] B. Xi, “Adversarial machine learning for cybersecurity and computer vision: Current developments and challenges,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 12, no. 5, pp. 1–16, 2020.
 - [87] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, “A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning,” *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, Sep. 2020.
 - [88] M. Z. Alom *et al.*, “A state-of-the-art survey on deep learning theory and architectures,” *Electronics (Switzerland)*, vol. 8, no. 3. MDPI AG, p. 292, 01-Mar-2019.
 - [89] P. Kumar and B. Raman, “A BERT based dual-channel explainable text emotion recognition system,” *Neural Networks*, vol. 150, pp. 392–407, Jun. 2022.

- [90] B. N. Supriya and C. B. Akki, “Sentiment prediction using enhanced xgboost and tailored random forest,” *Int. J. Comput. Digit. Syst.*, vol. 10, no. 1, pp. 191–199, Jan. 2021.
- [91] W. Peng, X. Hong, G. Zhao, and E. Cambria, “Adaptive Modality Distillation for Separable Multimodal Sentiment Analysis,” *IEEE Intell. Syst.*, vol. 36, no. 3, pp. 82–89, May 2021.
- [92] X. Huang, A. Dhall, R. Goecke, M. Pietikainen, and G. Zhao, “Multimodal Framework for Analyzing the Affect of a Group of People,” *IEEE Trans. Multimed.*, vol. 20, no. 10, pp. 2706–2721, Oct. 2018.
- [93] Q. A. Xu, V. Chang, and C. Jayne, “A systematic review of social media-based sentiment analysis: Emerging trends and challenges,” *Decis. Anal. J.*, vol. 3, p. 100073, Jun. 2022.
- [94] G. Fei, H. Li, and B. Liu, “Opinion Spam Detection in Social Networks,” in *Sentiment Analysis in Social Networks*, Elsevier Inc., 2017, pp. 141–156.
- [95] T. Mandhula, S. Pabboju, and N. Gugulotu, “Predicting the customer’s opinion on amazon products using selective memory architecture-based convolutional neural network,” *J. Supercomput.*, vol. 76, no. 8, pp. 5923–5947, Nov. 2020.
- [96] P. Korshunov and S. Marcel, “DeepFakes: A new threat to face recognition? Assessment and detection,” *arXiv*, pp. 1–5, 2018.
- [97] T. T. Nguyen *et al.*, “Deep learning for deepfakes creation and detection: A survey,” *Comput. Vis. Image Underst.*, vol. 223, pp. 1–16, 2022.
- [98] S. Cresci, “Detecting malicious social bots: Story of a never-ending clash,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12021 LNCS, pp. 77–88.

- [99] R. Zellers *et al.*, “Defending against neural fake news,” *Adv. Neural Inf. Process. Syst.*, vol. 32, May 2019.
- [100] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, “Deepfakes and beyond: A Survey of face manipulation and fake detection,” *Inf. Fusion*, vol. 64, pp. 131–148, Jan. 2020.
- [101] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, “FaceForensics++: Learning to detect manipulated facial images,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-Octob, pp. 1–11, Jan. 2019.
- [102] B. Alghamdi, Y. Xu, and J. Watson, “A Hybrid Approach for Detecting Spammers in Online Social Networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11233 LNCS, Springer International Publishing, 2018, pp. 189–198.
- [103] S. Kudugunta and E. Ferrara, “Deep neural networks for bot detection,” *Inf. Sci. (Ny)*, vol. 467, pp. 312–322, 2018.
- [104] R. S. Bhadoria, N. Bhoj, M. K. Srivastav, R. Kumar, and B. Raman, “A machine learning framework for security and privacy issues in building trust for social networking,” *Cluster Comput.*, pp. 1–24, Nov. 2022.
- [105] R. Ureña, G. Kou, Y. Dong, F. Chiclana, and E. Herrera-Viedma, “A review on trust propagation and opinion dynamics in social networks and group decision making frameworks,” *Inf. Sci. (Ny)*, vol. 478, pp. 461–475, Apr. 2019.
- [106] H. Mayadunna and L. Rupasinghe, “A Trust Evaluation Model for Online Social Networks,” in *National Information Technology Conference, NITC 2018*, 2018.
- [107] J. Wang, K. Qiao, and Z. Zhang, “Trust evaluation based on evidence theory in online social networks,” *Int. J. Distrib. Sens. Networks*, vol. 14, no. 10, p.

155014771879462, Oct. 2018.

- [108] C. Tong *et al.*, “A shilling attack detector based on convolutional neural network for collaborative recommender system in social aware network,” *Comput. J.*, vol. 61, no. 7, pp. 949–958, 2018.
- [109] X. Yang, M. Dong, X. Chen, and K. Ota, “Recommender System-Based Diffusion Inferring for Open Social Networks,” *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 1, pp. 24–34, Feb. 2020.
- [110] M. Ahsan and T. P. Sharma, “Spams classification and their diffusibility prediction on Twitter through sentiment and topic models,” *Int. J. Comput. Appl.*, vol. 44, no. 4, pp. 365–375, 2022.
- [111] C. Shao, G. L. Ciampaglia, O. Varol, K. C. Yang, A. Flammini, and F. Menczer, “The spread of low-credibility content by social bots,” *Nat. Commun.*, vol. 9, no. 1, p. 4787, Nov. 2018.
- [112] S. Cresci, M. Petrocchi, A. Spognardi, and S. Tognazzi, “Better safe than sorry: An adversarial approach to improve social bot detection,” 2019.
- [113] S. Sedhai and A. Sun, “Hspam14: A collection of 14 million tweets for hashtag-oriented spam research,” in *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 223–232.
- [114] S. Cresci, “A decade of social bot detection,” *Commun. ACM*, vol. 63, no. 10, pp. 72–83, Sep. 2020.
- [115] T. Q. Duong, J. A. Ansere, B. Narottama, V. Sharma, O. A. Dobre, and H. Shin, “Quantum-Inspired Machine Learning for 6G: Fundamentals, Security, Resource Allocations, Challenges, and Future Research Directions,” *IEEE Open J. Veh. Technol.*, vol. 3, pp. 375–387, 2022.

- [116] S. Namasudra, G. C. Deka, P. Johri, M. Hosseinpour, and A. H. Gandomi, “The Revolution of Blockchain: State-of-the-Art and Research Challenges,” *Arch. Comput. Methods Eng.*, vol. 1, p. 3, May 2020.
- [117] T. W. Jing and R. K. Murugesan, “A Theoretical Framework to Build Trust and Prevent Fake News in Social Media Using Blockchain,” in *International Conference of Reliable Information and Communication Technology*, 2018, vol. 2, pp. 139–150.
- [118] H. R. Hasan and K. Salah, “Combating Deepfake Videos Using Blockchain and Smart Contracts,” *IEEE Access*, vol. PP, no. c, p. 1, 2019.
- [119] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, “6 million spam tweets: A large ground truth for timely Twitter spam detection,” in *IEEE International Conference on Communications*, 2015, vol. 2015-September.
- [120] T. Wu, S. Wen, Y. Xiang, and W. Zhou, “Twitter spam detection: Survey of new approaches and comparative study,” *Comput. Secur.*, vol. 76, pp. 265–284, Jul. 2018.
- [121] T. Wu, S. Liu, J. Zhang, and Y. Xiang, “Twitter spam detection based on deep learning,” *ACM Int. Conf. Proceeding Ser.*, 2017.
- [122] S. Webb, J. Caverlee, and C. Pu, “Social Honeypots: Making Friends With A Spammer Near You. Denial of Information Project View project Synthetix View project Social Honeypots: Making Friends With A Spammer Near You,” 2008.
- [123] K. Lee, J. Caverlee, and S. Webb, “Uncovering social spammers: Social honeypots + machine learning,” *SIGIR 2010 Proc. - 33rd Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, no. January 2010, pp. 435–442, 2010.
- [124] H. Katakwar, P. Aggarwal, Z. Maqbool, and V. Dutt, “Influence of Network Size on Adversarial Decisions in a Deception Game Involving Honeypots,” *Front.*

Psychol., vol. 11, p. 2385, Sep. 2020.

- [125] P. Hayati, K. Chai, V. Potdar, and A. Talevski, “HoneySpam 2.0: Profiling web spambot behaviour,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5925 LNAI, pp. 335–344, 2009.
- [126] G. Stringhini, C. Kruegel, and G. Vigna, “Detecting spammers on social networks,” in *Proceedings - Annual Computer Security Applications Conference, ACSAC*, 2010, pp. 1–9.
- [127] A. Paradise *et al.*, “Creation and Management of Social Network Honeypots for Detecting Targeted Cyber Attacks,” *IEEE Trans. Comput. Soc. Syst.*, vol. 4, no. 3, pp. 65–79, 2017.
- [128] L. A. Cornelissen, R. J. Barnett, M. A. M. Kepa, D. Loebenberg-Novitzkas, and J. Jordaan, “Deploying South African social honeypots on twitter,” *ACM Int. Conf. Proceeding Ser.*, pp. 179–187, 2018.
- [129] Y. Zhang, H. Zhang, X. Yuan, and N. F. Tzeng, “Pseudo-Honeypot: Toward Efficient and Scalable Spam Sniffer,” *Proc. - 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Networks, DSN 2019*, pp. 435–446, 2019.
- [130] J. Liu and R. Danait, “Instance Selection in the Projected High Dimensional Feature Space for SVM,” *2018 3rd IEEE Int. Conf. Image, Vis. Comput. ICIVC 2018*, pp. 875–879, Oct. 2018.
- [131] H. Ahmed, I. Traore, and S. Saad, “Detecting opinion spams and fake news using text classification,” *Secur. Priv.*, vol. 1, no. 1, p. e9, 2018.
- [132] D. M. Beskow and K. M. Carley, “Bot conversations are different: Leveraging network metrics for bot detection in Twitter,” *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018*. pp. 825–832, 2018.

- [133] X. Zheng, Z. Zeng, Z. Chen, Y. Yu, and C. Rong, "Detecting spammers on social networks," *Neurocomputing*, vol. 159, no. 1, pp. 27–34, 2015.
- [134] S. Dutta, S. Ghatak, R. Dey, A. K. Das, and S. Ghosh, "Attribute selection for improving spam classification in online social networks: a rough set theory-based approach," *Soc. Netw. Anal. Min.*, vol. 8, no. 1, 2018.
- [135] M. K. Sohrabi and F. Karimi, "A Feature Selection Approach to Detect Spam in the Facebook Social Network," *Arab. J. Sci. Eng.*, vol. 43, no. 2, pp. 949–958, 2018.
- [136] D. Perna and A. Tagarelli, "Learning to Rank Social Bots," pp. 183–191, 2018.
- [137] A. M. Al-Zoubi, H. Faris, J. Alqatawna, and M. A. Hassonah, "Evolving Support Vector Machines using Whale Optimization Algorithm for spam profiles detection on online social networks in different lingual contexts," *Knowledge-Based Syst.*, vol. 153, pp. 91–104, 2018.
- [138] X. Ban, C. Chen, S. Liu, Y. Wang, and J. Zhang, "Deep-learnt features for Twitter spam detection," *2018 Int. Symp. Secur. Priv. Soc. Networks Big Data*, pp. 208–212, 2019.
- [139] S. R. Sahoo and B. B. Gupta, "Classification of spammer and nonspammer content in online social network using genetic algorithm-based feature selection," *Enterp. Inf. Syst.*, vol. 14, no. 5, pp. 710–736, May 2020.
- [140] S. R. Sahoo and B. B. Gupta, "Hybrid approach for detection of malicious profiles in twitter," *Comput. Electr. Eng.*, vol. 76, pp. 65–81, 2019.
- [141] Y. Hao and F. Zhang, "An unsupervised detection method for shilling attacks based on deep learning and community detection," *Soft Comput.*, vol. 25, no. 1, pp. 477–494, 2021.

- [142] H. Gupta, M. S. Jamal, S. Madisetty, and M. S. Desarkar, "A framework for real-time spam detection in Twitter," *2018 10th Int. Conf. Commun. Syst. Networks, COMSNETS 2018*, vol. 2018-Janua, no. January, pp. 380–383, 2018.
- [143] R. Narayan, J. K. Rout, and S. K. Jena, "Review Spam Detection Using Opinion Mining," pp. E3–E3, 2018.
- [144] S. Sedhai and A. Sun, "Semi-Supervised Spam Detection in Twitter Stream," *IEEE Trans. Comput. Soc. Syst.*, vol. 5, no. 1, pp. 169–175, 2018.
- [145] C. M. Yilmaz and A. O. Durahim, "SPR2EP: A semi-supervised spam review detection framework," *Proc. 2018 IEEE/ACM Int. Conf. Adv. Soc. Networks Anal. Mining, ASONAM 2018*, pp. 306–313, 2018.
- [146] A. Barushka and P. Hajek, *Spam filtering in social networks using regularized deep neural networks with ensemble learning*, vol. 519. Springer International Publishing, 2018.
- [147] G. Jain, M. Sharma, and B. Agarwal, "Spam Detection on Social Media Using Semantic Convolutional Neural Network," *Int. J. Knowl. Discov. Bioinforma.*, vol. 8, no. 1, pp. 12–26, 2018.
- [148] B. Feng, Q. Fu, M. Dong, D. Guo, and Q. Li, "Multistage and Elastic Spam Detection in Mobile Social Networks through Deep Learning," *IEEE Netw.*, vol. 32, no. 4, pp. 15–21, 2018.
- [149] S. G. Selvaganapathy, M. Nivaashini, and H. P. Natarajan, "Deep belief network based detection and categorization of malicious URLs," *Inf. Secur. J.*, vol. 27, no. 3, pp. 145–161, 2018.
- [150] V. Singh, A. Varshney, S. S. Akhtar, D. Vijay, and M. Shrivastava, "Aggression Detection on Social Media Text Using Deep Neural Networks," *2nd Work. Abus. Lang. Online - Proc. Work. co-located with EMNLP 2018*, no. October, pp. 43–50,

2018.

- [151] B. Wu, L. Liu, Y. Yang, K. Zheng, and X. Wang, "Using improved conditional generative adversarial networks to detect social bots on Twitter," *IEEE Access*, vol. 8, pp. 36664–36680, 2020.
- [152] Y. Wu, Y. Fang, S. Shang, J. Jin, L. Wei, and H. Wang, "A novel framework for detecting social bots with deep neural networks and active learning," *Knowledge-Based Syst.*, vol. 211, p. 106525, Jan. 2021.
- [153] H. Tajalizadeh and R. Boostani, "A Novel Stream Clustering Framework for Spam Detection in Twitter," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 3, pp. 525–534, 2019.
- [154] T. Xia, "A constant time complexity spam detection algorithm for boosting throughput on rule-based filtering systems," *IEEE Access*, vol. 8, pp. 82653–82661, 2020.
- [155] K. Liu, B. Fang, and Y. Zhang, "Detecting tag spam in social tagging systems with collaborative knowledge," *6th Int. Conf. Fuzzy Syst. Knowl. Discov. FSKD 2009*, vol. 7, pp. 427–431, 2009.
- [156] X. Zhang, H. Bai, and W. Liang, "A social spam detection framework via semi-supervised learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9794, no. 61272374, pp. 214–226, 2016.
- [157] S. Saumya and J. P. Singh, "Detection of spam reviews: a sentiment analysis approach," *CSI Trans. ICT*, vol. 6, no. 2, pp. 137–148, May 2018.
- [158] A. Mehmood, B. W. On, I. Lee, I. Ashraf, and G. Sang Choi, "Spam comments prediction using stacking with ensemble learning," *J. Phys. Conf. Ser.*, vol. 933, no. 1, 2018.

- [159] I. J. Cruickshank and K. M. Carley, “Characterizing communities of hashtag usage on twitter during the 2020 COVID-19 pandemic by multi-view clustering,” *Appl. Netw. Sci.*, vol. 5, no. 1, 2020.
- [160] H. Shen, F. Ma, X. Zhang, L. Zong, X. Liu, and W. Liang, “Discovering social spammers from multiple views,” *Neurocomputing*, vol. 225, no. October 2016, pp. 49–57, 2017.
- [161] Q. Fu, B. Feng, D. Guo, and Q. Li, “Combating the evolving spammers in online social networks,” *Comput. Secur.*, vol. 72, pp. 60–73, 2018.
- [162] A. Singh and S. Batra, “Ensemble based spam detection in social IoT using probabilistic data structures,” *Futur. Gener. Comput. Syst.*, vol. 81, pp. 359–371, 2018.
- [163] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, and T. Chakraborty, “Collective Classification of Spam Campaigners on Twitter: A Hierarchical Meta-Path Based Approach,” 2018.
- [164] A. D. Kumar, V. R., and S. KP, “DeepImageSpam: Deep Learning based Image Spam Detection,” 2018.
- [165] Y. Li, M. C. Chang, and S. Lyu, “In Ictu Oculi: Exposing AI created fake videos by detecting eye blinking,” in *10th IEEE International Workshop on Information Forensics and Security, WIFS 2018*, 2019.
- [166] D. Guera and E. J. Delp, “Deepfake Video Detection Using Recurrent Neural Networks,” *Proc. AVSS 2018 - 2018 15th IEEE Int. Conf. Adv. Video Signal-Based Surveill.*, 2018.
- [167] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, “Recurrent Convolutional Strategies for Face Manipulation Detection in Videos,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern*

Recognition Workshops ., 2019, pp. 80–87.

- [168] M. Singh, D. Bansal, and S. Sofat, “Who is Who on Twitter–Spammer, Fake or Compromised Account? A Tool to Reveal True Identity in Real-Time,” *Cybern. Syst.*, vol. 49, no. 1, pp. 1–25, 2018.
- [169] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, “BotOrNot: A system to evaluate social bots,” 2016, pp. 273–274.
- [170] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, “Online Human Bot Interaction,” *Proc. Elev. Int. AAAI Conf. Web Soc. Media (ICWSM 2017)*, no. Icwsn, pp. 280–289, 2017.
- [171] D. M. Beskow and K. M. Carley, “Bot-hunter: A tiered approach to detecting and characterizing automated activity on twitter,” *2018 Int. Conf. Soc. Comput. Behav. Model. Predict. Behav. Represent. Model. Simulation, BRiMS 2018*, no. July, pp. 1–8, 2018.
- [172] G. Lingam, R. R. Rout, and D. V. L. N. Somayajulu, “Adaptive deep Q-learning model for detecting social bots and influential users in online social networks,” *Applied Intelligence*. 2019.
- [173] P. V. Bindu, R. Mishra, and P. S. Thilagam, “Discovering spammer communities in twitter,” *J. Intell. Inf. Syst.*, vol. 51, no. 3, pp. 503–527, 2018.
- [174] E. Van Der Walt and J. Eloff, “Using Machine Learning to Detect Fake Identities: Bots vs Humans,” *IEEE Access*, vol. 6, pp. 6540–6549, 2018.
- [175] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, “Social Fingerprinting: Detection of Spambot Groups Through DNA-Inspired Behavioral Modeling,” *IEEE Trans. Dependable Secur. Comput.*, vol. 15, no. 4, pp. 561–576, 2018.

- [176] S. Jeong, G. Noh, H. Oh, and C. kwon Kim, "Follow spam detection based on cascaded social information," *Inf. Sci. (Ny)*, vol. 369, pp. 481–499, 2016.
- [177] M. Fazil and M. Abulaish, "A Hybrid Approach for Detecting Automated Spammers in Twitter," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 11, pp. 2707–2719, 2018.
- [178] C. Xiao, D. M. Freeman, and T. Hwa, "Detecting Clusters of Fake Accounts in Online Social Networks," pp. 91–101, 2015.
- [179] M. Al-Qurishi, M. Alrubaian, S. M. M. Rahman, A. Alamri, and M. M. Hassan, "A prediction system of Sybil attack in social network using deep-regression model," *Futur. Gener. Comput. Syst.*, vol. 87, pp. 743–753, 2018.
- [180] O. de Lima, S. Franklin, S. Basu, B. Karwoski, and A. George, "Deepfake Detection using Spatiotemporal Convolutional Networks," *arXiv*, 2020.
- [181] J. Hernandez-Ortega, R. Tolosana, J. Fierrez, and A. Morales, "DeepFakesON-Phys: DeepFakes Detection based on Heart Rate Estimation," *arXiv*, 2020.
- [182] D. M. Montserrat *et al.*, "Deepfakes detection with automatic face weighting," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2020-June, pp. 2851–2859, 2020.
- [183] Y. Li and S. Lyu, "Exposing DeepFake Videos by Detecting FaceWarping Artifacts," *arXiv*, Nov. 2019.
- [184] T. Jung, S. Kim, and K. Kim, "DeepVision: Deepfakes Detection Using Human Eye Blinking Pattern," *IEEE Access*, vol. 8, pp. 83144–83154, 2020.
- [185] H. Qi *et al.*, "DeepRhythm: Exposing DeepFakes with Attentional Visual Heartbeat Rhythms," *MM 2020 - Proc. 28th ACM Int. Conf. Multimed.*, pp. 1318–1327, Jun. 2020.

- [186] P. R. B. Satya, K. Lee, D. Lee, T. Tran, and J. Zhang, “Uncovering fake likers in online social networks,” in *International Conference on Information and Knowledge Management, Proceedings*, 2016, vol. 24-28-Octo, pp. 2365–2370.
- [187] S. Rayana and L. Akoglu, “Collective opinion spam detection: Bridging review networks and metadata,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, vol. 2015-Augus, pp. 985–994.
- [188] A. Mukherjee, V. Venkataraman, and B. Liu, “What Yelp fake review filter might be doing?,” *Proc. Int. AAAI Conf. Web Soc. Media*, vol. 7, no. 1, pp. 409–418, 2011.
- [189] M. Hopkins, E. Reeber, G. Forman, and J. Suermondt, “UCI Machine Learning Repository: Spambase Data Set,” 2016. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/spambase>. [Accessed: 14-Nov-2020].
- [190] R. Mohammad, L. McCluskey, and F. Thabtah, “UCI Machine Learning Repository: Phishing Websites Data Set,” 2016. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/phishing+websites>. [Accessed: 14-Nov-2020].
- [191] MDL, “MDL: Malware Domian List,” 2009. [Online]. Available: <http://www.malwaredomainlist.com/>. [Accessed: 14-Sep-2020].
- [192] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, “Finding Deceptive Opinion Spam by Any Stretch of the Imagination,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 309–319.
- [193] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, “Detecting spammers on Twitter,” in *7th Annual Collaboration, Electronic Messaging, Anti-Abuse and*

Spam Conference, CEAS 2010, 2010.

- [194] F. Benevenuto, T. Rodrigues, J. Almeida, M. Gonçalves, and V. Almeida, “Detecting spammers and content promoters in online video social networks,” in *Proceedings - IEEE INFOCOM*, 2009.
- [195] H. Costa, F. Benevenuto, and L. H. C. Merschmann, “Detecting tip spam in location-based social networks,” *Proc. ACM Symp. Appl. Comput.*, pp. 724–729, 2013.
- [196] H. Costa, L. H. C. Merschmann, F. Barth, and F. Benevenuto, “Pollution, bad-mouthing, and local marketing: The underground of location-based social networks,” *Inf. Sci. (Ny)*, vol. 279, pp. 123–137, Sep. 2014.
- [197] T. A. Almeida and T. C. Alberto, “UCI Machine Learning Repository: YouTube Spam Collection Data Set,” 2015. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/YouTube+Spam+Collection>. [Accessed: 20-Nov-2020].
- [198] M. Ott, C. Cardie, and J. T. Hancock, “Negative Deceptive Opinion Spam,” Association for Computational Linguistics, 2013.
- [199] T. A. Almeida and J. M. Ga. Hidalgo, “UCI Machine Learning Repository: SMS Spam Collection Data Set,” 2016. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection>. [Accessed: 14-Sep-2020].
- [200] Kaggle.com, “TalkingData AdTracking Fraud Detection Challenge | Kaggle,” 2018. [Online]. Available: <https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data>. [Accessed: 12-Aug-2020].
- [201] R. Kumar, A. N. Reganti, A. Bhatia, T. Maheshwari, and B. Rao, *Aggression-annotated Corpus of Hindi-English Code-mixed Data*. Miyazaki, Japan: European

Language Resources Association (ELRA), 2018.

- [202] C. M. Lai, M. H. Chen, E. Kristiani, V. K. Verma, and C. T. Yang, “Fake News Classification Based on Content Level Features,” *Appl. Sci.*, vol. 12, no. 3, 2022.
- [203] P. K. Verma, P. Agrawal, I. Amorim, and R. Prodan, “WELFake: Word Embedding over Linguistic Features for Fake News Detection,” *IEEE Trans. Comput. Soc. Syst.*, vol. 8, no. 4, pp. 881–893, Aug. 2021.
- [204] U. Naseem, S. Khalid Khan, and M. Prasad, “A Comprehensive Survey on Word Representation Models: From Classical to State-of-the-Art Word Representation Language Models,” *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 20, 2021.
- [205] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.
- [206] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality arXiv : 1310 . 4546v1 [cs . CL] 16 Oct 2013,” *arXiv Prepr. arXiv1310.4546*, vol. cs.CL, pp. 1–9, 2013.
- [207] Q. Jiao and S. Zhang, “A Brief Survey of Word Embedding and Its Recent Development,” *IAEAC 2021 - IEEE 5th Adv. Inf. Technol. Electron. Autom. Control Conf.*, vol. 2021, pp. 1697–1701, 2021.
- [208] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation,” in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014, pp. 1532–1543.
- [209] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146,

Jul. 2016.

- [210] T. Mikolov, E. Grave, P. Bojanowski, C. Puhersch, and A. Joulin, “Advances in Pre-Training Distributed Word Representations,” *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 52–55, Dec. 2017.
- [211] H. Nguyen and R. Derakhshani, “Eyebrow Recognition for Identifying Deepfake Videos,” *Lect. Notes Informatics (LNI), Proc. - Ser. Gesellschaft fur Inform.*, vol. P-306, pp. 199–206, 2020.
- [212] P. Neekhara, B. Dolhansky, J. Bitton, and C. C. Ferrer, “Adversarial threats to deepfake detection: A practical perspective,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, pp. 923–932, 2021.
- [213] L. Jiang, R. Li, W. Wu, C. Qian, and C. C. Loy, “Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 2886–2895, 2020.
- [214] A. Gandhi and S. Jain, “Adversarial Perturbations Fool Deepfake Detectors,” *Proc. Int. Jt. Conf. Neural Networks*, no. Ijcn, 2020.
- [215] E. Tursman, M. George, S. Kamara, and J. Tompkin, “Towards untrusted social video verification to combat deepfakes via face geometry consistency,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2020-June, pp. 2784–2793, 2020.
- [216] X. Yang, Y. Li, and S. Lyu, “Exposing Deep Fakes Using Inconsistent Head Poses,” *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2019-May, pp. 8261–8265.
- [217] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, “The Deepfake Detection Challenge (DFDC) Preview Dataset,” *arXiv*, Oct. 2019.

- [218] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “CVML Celeb-DF dataset,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [219] Y. Zhu, X. Wang, E. Zhong, N. N. Liu, H. Li, and Q. Yang, “Discovering spammers in social networks,” *Proc. Natl. Conf. Artif. Intell.*, vol. 1, pp. 171–177, 2012.
- [220] S. Rathore, V. Loia, and J. H. Park, “SpamSpotter: An efficient spammer detection framework based on intelligent decision support system on Facebook,” *Appl. Soft Comput. J.*, vol. 67, pp. 920–932, 2018.
- [221] P. A. K. Lorimer, V. M. F. Diec, and B. Kantarci, “Participatory detection of identity theft on mobile social platforms,” in *2017 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2017 - Proceedings*, 2018, vol. 2018-Janua, pp. 833–837.
- [222] L. Fu, “A comprehensive framework for detecting sybils and spammers on social networks,” *ICEIS 2018 - Proc. 20th Int. Conf. Enterp. Inf. Syst.*, vol. 1, no. Iceis 2018, pp. 229–236, 2018.
- [223] P. Gao, B. Wang, N. Z. Gong, S. R. Kulkarni, K. Thomas, and P. Mittal, “SybilFuse: Combining local attributes with global structure to perform robust sybil detection,” *arXiv*, 2018.
- [224] K. Lee, B. Eoff, and J. Caverlee, “Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter,” *Proc. Int. AAAI Conf. Web Soc. Media*, vol. 5, no. 1, pp. 185–192, 2021.
- [225] H. Kwak, C. Lee, H. Park, and S. Moon, “What is Twitter, a social network or a news media?,” in *Proceedings of the 19th International Conference on World Wide Web, WWW ’10*, 2010, pp. 591–600.
- [226] O. Varol, E. Ferrara, F. Menczer, and A. Flammini, “Early detection of promoted

- campaigns on social media,” *EPJ Data Sci.*, vol. 6, no. 1, Dec. 2017.
- [227] Z. Gilani, L. Wang, J. Crowcroft, M. Almeida, and R. Farahbakhsh, “Stweeler: A Framework for Twitter Bot Analysis,” in *International World Wide Web Conference Committee (IW3C2)*, 2016, pp. 37–38.
- [228] H. Kaur, S. U. Ahsaan, B. Alankar, and V. Chang, “A Proposed Sentiment Analysis Deep Learning Algorithm for Analyzing COVID-19 Tweets,” *Inf. Syst. Front.*, vol. 23, no. 6, pp. 1417–1429, Dec. 2021.
- [229] S. Rao, A. K. Verma, and T. Bhatia, “A review on social spam detection: Challenges, open issues, and future directions,” *Expert Syst. Appl.*, vol. 186, p. 115742, Aug. 2021.
- [230] M. Tolba, S. Ouadfel, and S. Meshoul, “Hybrid ensemble approaches to online harassment detection in highly imbalanced data,” *Expert Syst. Appl.*, vol. 175, no. January, p. 114751, 2021.
- [231] S. Aiyar and N. P. Shetty, “N-Gram Assisted Youtube Spam Comment Detection,” *Procedia Comput. Sci.*, vol. 132, pp. 174–182, Jan. 2018.
- [232] T. C. Alberto, J. V. Lochter, and T. A. Almeida, “TubeSpam: Comment spam filtering on YouTube,” *Proc. - 2015 IEEE 14th Int. Conf. Mach. Learn. Appl. ICMLA 2015*, no. January, pp. 138–143, 2016.
- [233] B. Kardaş, I. E. Bayar, T. Özyer, and R. Alhajj, “Detecting spam tweets using machine learning and effective preprocessing,” *Proc. 2021 IEEE/ACM Int. Conf. Adv. Soc. Networks Anal. Mining, ASONAM 2021*, pp. 393–398, Nov. 2021.
- [234] K. W. P. Chawla, N.V., Bowyer, K.W., Hall, L.O., “SMOTE: Synthetic Minority Over-Sampling Technique. Journal of Artificial Intelligence Research,” *Journal of Artificial Intelligence Research*, 2002. [Online]. Available: <https://dl.acm.org/doi/10.5555/1622407.1622416>. [Accessed: 01-Jul-2022].

- [235] H. Ghaderi Zefrehi and H. Altınçay, “Imbalance learning using heterogeneous ensembles,” *Expert Syst. Appl.*, vol. 142, p. 113005, Mar. 2020.
- [236] F. Sağlam and M. A. Cengiz, “A novel SMOTE-based resampling technique through noise detection and the boosting procedure,” *Expert Syst. Appl.*, vol. 200, p. 117023, Aug. 2022.
- [237] Z. Wang, C. Wu, K. Zheng, X. Niu, and X. Wang, “SMOTETomek-Based Resampling for Personality Recognition,” *IEEE Access*, vol. 7, pp. 129678–129689, 2019.
- [238] X. Tao *et al.*, “SVDD-based weighted oversampling technique for imbalanced and overlapped dataset learning,” *Inf. Sci. (Ny)*, vol. 588, pp. 13–51, Apr. 2022.
- [239] N. Ghatasheh *et al.*, “Cost-sensitive ensemble methods for bankruptcy prediction in a highly imbalanced data distribution: a real case from the Spanish market,” *Prog. Artif. Intell.*, vol. 9, no. 4, pp. 361–375, Mar. 2020.
- [240] M. Gu, “Ling-Spam Dataset | Kaggle,” *Kaggle.com*, 2019. [Online]. Available: <https://www.kaggle.com/mandygu/lingspam-dataset/metadata>. [Accessed: 03-Mar-2022].
- [241] P. Patwa *et al.*, “Fighting an Infodemic: COVID-19 Fake News Dataset,” *Commun. Comput. Inf. Sci.*, vol. 1402 CCIS, pp. 21–29, 2021.
- [242] K. Agarwal and T. Kumar, “Email Spam Detection Using Integrated Approach of Naïve Bayes and Particle Swarm Optimization,” *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. ICICCS 2018*, pp. 685–690, Mar. 2019.
- [243] Y. Albalawi, J. Buckley, and N. S. Nikolov, “Investigating the impact of pre-processing techniques and pre-trained word embeddings in detecting Arabic health information on social media,” *J. Big Data*, vol. 8, no. 1, 2021.

- [244] S. Madisetty and M. S. Desarkar, “A Neural Network-Based Ensemble Approach for Spam Detection in Twitter,” *IEEE Trans. Comput. Soc. Syst.*, vol. 5, no. 4, pp. 973–984, 2018.
- [245] G. Jain, M. Sharma, and B. Agarwal, “Spam detection in social media using convolutional and long short term memory neural network,” *Ann. Math. Artif. Intell.*, vol. 85, no. 1, pp. 21–44, Jan. 2019.
- [246] A. Barushka and P. Hajek, “Spam detection on social networks using cost-sensitive feature selection and ensemble-based regularized deep neural networks,” *Neural Comput. Appl.*, vol. 32, no. 9, pp. 4239–4257, May 2020.
- [247] J. A. Nasir, O. S. Khan, and I. Varlamis, “Fake news detection: A hybrid CNN-RNN based deep learning approach,” *Int. J. Inf. Manag. Data Insights*, vol. 1, no. 1, p. 100007, 2021.
- [248] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, “Deep Learning Based Text Classification: A Comprehensive Review,” vol. 1, no. 1, pp. 1–43, 2020.
- [249] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” Dec. 2014.
- [250] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553. Nature Publishing Group, pp. 436–444, 27-May-2015.
- [251] Z. Niu, G. Zhong, and H. Yu, “A review on the attention mechanism of deep learning,” *Neurocomputing*, vol. 452, pp. 48–62, Sep. 2021.
- [252] A. Vaswani *et al.*, “Attention is all you need,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 5999–6009, Jun. 2017.