

UNITEC INSTITUTE OF TECHNOLOGY

MASTER THESIS

**SMS PHISHING DETECTION USING MACHINE LEARNING
AND DEEP LEARNING TECHNIQUES**

Author: Pavan Hasti

Principal Supervisor: Bashar Barmada

Associate Supervisor: Soheil Pour

A thesis submitted in partial fulfilment of the requirements
for the degree of Master of Applied Technologies
in the
School of Computing, Electrical & Applied Technology

Date: 30 April 2025

ABSTRACT

Short Message Service (SMS) is still a vital communication tool in our daily life activities, even with the quick development of Internet protocol-based messaging services. An increasingly sophisticated cyber threat known as SMS phishing (smishing) has emerged in tandem with the rise in mobile device use. As a result, people are finding it hard to distinguish good messages from bad ones. The attackers' propensity for always developing methods has made smishing detection problematic for typical phishing detection techniques including heuristic, feature-based, rule-based, and blacklist approaches. The aim is to construct a New Zealand domain-specific SMS phishing detection system to overcome these difficulties and improve mobile system cybersecurity. The goal is to build an effective ML and DL-based model to accurately detect and categorize SMS smishing messages, addressing class imbalance and ensuring ethical data handling for improved cybersecurity. This study collects the dataset, which is the combination of SMS Smishing Collection from Kaggle and smishing messages from the New Zealand Department of Internal Affairs (DIA) anti-scam archive, ensuring relevance to the local context. The Pre-processing methods involved steps to manage missing and duplicated values, while checking label uniqueness, and performing text pre-processing and lemmatization, followed by label encoding. The dataset is balanced with SMOTE. Random Forest and XGBoost, CNN, RNN, and LSTM are some of the deep learning and machine learning classification models selected for their exceptional performance in text analysis. The models work well for detecting fake SMS messages in the setting of mobile communication networks. Accuracy, precision, recall, and F1 score were some of the important measures used to assess the models' performance. The result showed that the XGBoost classifier achieving a superior accuracy of 97.05% compared to other models. This study highlights the practical implications of smishing detection, particularly in real-world mobile communication systems, emphasizing the importance of integrating these models into mobile security applications. Additionally, the research discusses potential future work, including the integration of transformer-based models, the handling of model drift, and addressing adversarial concerns in dynamic environments.

Table of Contents

Abstract.....	ii
List of Figures.....	vi
List of Tables	viii
List of Abbreviations	ix
Chapter 1 Introduction	1
1.1 Research objectives	3
1.2 Main Research Questions or Hypotheses.....	4
1.3 Thesis organization.....	4
Chapter 2 Literature of Review.....	6
2.1 SMS Phishing in New Zealand Bank.....	6
2.2 Machine Learning and Deep Learning Techniques.....	8
2.3 Critical Comparisons of Existing Models	16
2.4 Research Gaps	16
2.5 Phishing Trends in New Zealand Compared to Global Trends	17
2.6 Tools and Techniques	17
2.6.1 Text Tokenization	17
2.6.2 Word2Vec Embedding.....	18
2.6.3 Data Balancing Using SMOTE	18
2.6.4 XGBoost-Classifer-Model.....	19
2.6.5 Random-Forest-Classifer-Model.....	20
2.6.6 Convolutional-Neural-Networks (CNN) Model	21
2.6.7 Recurrent-Neural-Networks (RNN) Model.....	23
2.6.8 Long-Short-Term-Memory (LSTM) Model.....	23
2.6.9 Simulation Tool	24
Chapter 3 Research Methodology.....	27
3.1 DATA PREPARATION AND EXPLORATION	28

3.1.1	Loading the Dataset.....	29
3.1.2	Extracting Short Messages:.....	29
3.1.3	Class Distribution.....	29
3.1.4	Pre-processing.....	29
3.2	Feature Extraction and Embeddings.....	31
3.3	Class Balancing with SMOTE.....	31
3.4	Data Splitting.....	32
3.5	Proposed architecture of deep learning and Machine learning models.....	32
3.5.1	Hyperparameter tuning of CNN, RNN, and LSTM.....	33
3.6	Computational Efficiency of models: Model Efficiency and Real-World Feasibility	36
3.6.1	Model justification.....	36
3.7	Model Evaluation Metrics for SMS Smishing Detection.....	37
Chapter 4	Experimental Results.....	38
4.1	Dataset Analysis Results.....	38
4.2	Classification Results.....	41
4.2.1	XGBoost classifier.....	41
4.2.2	Random Forest classifier.....	42
4.2.3	CNN Model.....	44
4.2.4	RNN Model.....	47
4.2.5	LSTM Model.....	49
4.3	Comparative analysis.....	51
4.3.1	Limitations of the Current Approach and Improvement.....	52
4.3.2	Impact of Misclassifications.....	53
4.4	Discussion.....	53
4.5	Critical evaluation of results.....	54
Chapter 5	Conclusion and Future Work.....	55

5.1	Conclusion.....	55
5.2	Limitations.....	57
5.3	Future work	57
	References	59

List of Figures

Figure 1. An illustration of SMOTE (Synthetic Minority Over-sampling Technique) applied to a classification scenario that does not achieve balance. SMOTE creates new instances of the minority class by interpolating between nearby samples (Jefferson, 2020).....	19
Figure 2. General architecture of XG boost classifier(T. Chen & Guestrin, 2016).....	20
Figure 3. Working of random forest classifier (Logunova, 2022).....	21
Figure 4. Fundamental architecture of the CNN model (Do et al., 2021).....	22
Figure 5. Working of RNN Model (Gupta et al., 2024)	23
Figure 6. The LSTM memory blocks structure (Su, 2020).	24
Figure 7 The Proposed Block diagram of the methodology to build and evaluate the ML/DL detection system for smishing	28
Figure 8: Model summary of CNN	35
Figure 9: Model summary of RNN	35
Figure 10: Model summary of LSTM	36
Figure 11. Distribution of Labels	38
Figure 12. Word cloud of Ham messages.....	39
Figure 13. Word cloud for Smish message.....	39
Figure 14. Distribution of labels before balancing.....	40
Figure 15. Distribution of labels after balancing.....	40
Figure 16. Classification report of XGBoost classifier Model.....	41
Figure 17. Confusion Matrix of XGBoost classifier	41
Figure 18. ROC Curve of XGBoost model	42
Figure 19. Classification report of Random Forest classifier.....	43
Figure 20. Confusion matrix of Random Forest classifier	43
Figure 21. Roc Curve of Random Forest Classifier	44
Figure 22. Model loss graph of CNN model	44
Figure 23. Model accuracy graph of train and test model.....	45
Figure 24. Classification report of CNN model	46
Figure 25. Confusion matrix of CNN model.....	46
Figure 26. ROC Curve of CNN model.....	46
Figure 27. RNN Model Loss graph of Train and test model.....	47

Figure 28. RNN Model Accuracy graph of Train and test model	47
Figure 29. Classification report of RNN model	48
Figure 30. Confusion matrix of RNN model.....	48
Figure 31 ROC curve of the RNN model.....	49
Figure 32. Loss graph of LSTM model.....	49
Figure 33. Accuracy graph of LSTM model	50
Figure 34. Confusion matrix of LSTM model	51
Figure 35. The LSTM models ROC curve	51

List of Tables

Table 1: Critical comparisons of existing models on ML/DL in SMS phishing detection	15
Table 2. XGBoost classifier Model	41
Table 3. Random Forest classifier Model.....	42
Table 4. Performance of CNN model.....	45
Table 5. RNN model performance.....	48
Table 6. LSTM Model Performance Results	50
Table 7. Comparison of the ML and DL models under study	52

List of Abbreviations

Abbreviation	Full Form
SMS	Short Message Service
ML	Machine Learning
NLP	Natural Language Processing
SMOTE	Synthetic Minority Over-Sampling Technique
CNN	Convolutional Neural Network
RNNs	Recurrent Neural Networks
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
DNNs	Deep Neural Networks
SVM	Support Vector Machine
DL	Deep Learning
ASASYN	Adaptive Synthetic Sampling

Chapter 1 Introduction

Texting has become an integral part of modern life, with billions of individuals using Short Message Service (SMS) for both personal and work-related conversations. Unfortunately, a new mobile security risk known as SMS phishing (smishing) has emerged in recent years, even though SMS is among the most widely utilised communication technologies in use today (Mishra & Soni, 2019). The use of SMS assaults by cybercriminals allows them to trick individuals into divulging sensitive information. Smishing represents a merger of SMS technology and phishing techniques. attackers may use deceptive messages pretending to be from known sources that contain dangerous URLs leading victims to fake websites (Arab & Sohrabi, 2017). Another example of fraudulent smishing messages is to simulate a bank notification that tricks victims into clicking a link to supply their account information by claiming that need to unlock their debit card or receive a special deal (Delany et al., 2012); Airehrour, Nair and Madanian, 2018).

The growing number of smishing attacks represents a severe cybersecurity issue because mobile device users tend to underestimate the dangers of SMS-based scams. SMS attacks are harder to detect because people trust communication through text messages compared to familiar phishing attempts through email. Mobile devices remain susceptible to these attacks because security features typically present in desktop computers are not available on mobile platforms, and diversity of people using mobile phones is much wider than that who use computers (Akande et al., 2023).

The traditional phishing technique adopts its predatory methods through Smishing by exploiting digital communication channels that use emails and other forms. Text messaging gained popularity, so phishing attacks transformed into smishing, which exploits the immediate and personal SMS communication features. The adaptation of deceptive text messages into fraudulent scams represents the concept of smishing as criminals manipulate SMS communication to steal victim information or deliver malware. The implementation of machine learning advancements produces superior phishing detection through improved data analysis and decreased false positive rates (Mishra & Soni, 2023). The detection of smishing messages depends on analysing characteristics that reveal potential fraudulent intent. Message content-based features analyse the presence of distinct keywords and recurring phrases together with patterns that are typically found in scam text messages. The features help ML models to detect

malicious messages that include unknown and false links together with unnecessary pressure to act immediately. The evaluation of sender-based features includes checking the phone number of the sender and message frequency together with the reputation of the sender as a scam source. URL-based features examine SMS links including shortened URLs as well as IP addresses instead of domain names associated with deceptive web addresses that impersonate legitimate sites. The structural characteristics of messages that use excessive capitalization and special characters coupled with irregular punctuation offer valuable evidence to detect smishing attempts. Behavioural features evaluate how messages get distributed through an analysis of three elements: the number of messages sent in each period and the repetition rate of matching messages sent to various recipients. Smishing attack identification reaches higher effectiveness when trained models analyse these attack features through NLP and ML processing techniques (Mishra & Soni, 2022) (Hameed, 2021)(Manikkannan, 2023).

The primary objective of this project is to create a machine learning and deep learning-based SMS phishing detection system tailored to New Zealand. The research creates its localized dataset by blending the Kaggle ¹²SMS Smishing Collection with real SMS phishes obtained from the New Zealand Department of Internal Affairs (DIA). This research develops an extensive dataset for SMS phishing detection through text normalization techniques, which combine tokenization and stop-word filtering followed by class-imbalanced rectification methods based on oversampling and under sampling procedure before running multiple ML and DL models to find optimal evaluation metrics and performing detailed model assessment procedures. This research aims to find the superior models from ML and DL algorithms for detecting SMS phishing content because New Zealand faces increasing smishing threats in its digital landscape. This will help stopping the financial losses, safeguarding user data, and expanding global understanding of phishing identification methods.

The successful implementation of research goals requires an organized methodology framework. The research adopts a systematic method which starts by building a New Zealand-specific dataset through the combination of Kaggle SMS Smishing Collection with authentic examples from the New Zealand DIA. The dataset undergoes pre-processing methods, which tokenize words while removing stop words and normalizing text, to enhance its quality along with model accuracy. The methodology requires specific attention to class imbalance through multiple balancing methods, which enable the model to perform accurately on phishing and

¹ <https://www.kaggle.com/datasets/galactus007/sms-smishing-collection-data-set>

² <https://www.kaggle.com/datasets/galactus007/sms-smishing-collection-data-set>

legitimate SMS identification. Model performance is evaluated through precision along with recall and F1-score and ROC-AUC as the selected metrics for validation of the approach's effectiveness.

The experimental section carries out methodological implementation, followed by practical testing of the proposed approaches. Following training on the generated dataset, the study assesses the efficacy of several ML and DL models in detecting SMS phishing attempts. This part explores how different class balancing techniques affect model accuracy when identifying phishing and legitimate messages within the system. For realistic SMS phishing detection systems in New Zealand, metrics including ROC-AUC, F1-score, recall, and precision are utilized to choose the optimal model.

1.1 RESEARCH OBJECTIVES

This project's overarching goal is to build a New Zealand-specific SMS phishing detection system by integrating data from the New Zealand DIA with the Kaggle SMS Smishing Collection. The goal is to build an effective ML and DL-based model to accurately detect and categorize SMS phishing messages, addressing class imbalance and ensuring ethical data handling for improved cybersecurity.

The following research objectives are as follows:

- **To develop a New Zealand-specific SMS phishing dataset:** Construct a dataset tailored for phishing detection by combining the Kaggle SMS Smishing Collection with examples by the New Zealand DIA.
- **To apply data pre-processing techniques for text normalization:** Preprocessing the data involves activities like tokenization, Stop Words removal, removing numbers and special characters, and stemming and lemmatization for enhanced model learning using ML and DL models.
- **To address the class imbalance in the dataset:** Use oversampling and under-sampling techniques to help balance the given dataset in the hope of gaining higher accuracy for the model at identifying both Phishing and legitimate Short Message Service SMS messages.
- **To evaluate various machine learning and deep learning models:** Explore ML and DL models to help in detecting SMS phishing (smishing).
- **To optimize the model evaluation metrics:** As for the evaluation process of the trained models, three well-known performance indicators – precision, accuracy, recall – should

be used, F1-score, and ROC-AUC should be applied for selecting the best performing model.

- **To provide a comprehensive analysis of model performance:** This is done by comparing a performance of a various models using evaluation metrics to discover the suitability of each in detecting the SMS phishing messages.

1.2 MAIN RESEARCH QUESTIONS OR HYPOTHESES.

The following research questions are as follows:

- How effective are ML and DL models in detecting SMS phishing in New Zealand's mobile system?
- What impact does class imbalance have, and how can SMOTE address this in phishing detection?
- Which ML and DL models perform best for SMS phishing detection in New Zealand?
- How can ethical data handling be maintained in SMS phishing detection?
- What real-world challenges (model drift, adversarial attacks) could affect SMS phishing detection, and how can they be mitigated?

The following hypotheses are as follows:

- Deep learning models will outperform traditional ML models in SMS phishing detection.
- SMOTE will improve model performance by addressing class imbalance.
- DL models will generalize better than ML models for real-world SMS phishing data.
- Ethical data handling will not affect model performance but must be considered in deployment.
- Model drift and adversarial attacks will pose challenges, requiring mitigation strategies for stable performance.

1.3 THESIS ORGANIZATION

The report follows a multi-chapter organization. Chapter 2 of the literature review provides a comprehensive overview of research on ML and DL methods for smishing detection via text message. Chapter 3, "Research Design and Methodology:" describes the study's methodology and design in comprehensive detail. It describes the process of constructing a new dataset for the specific context of New Zealand containing SMS phishing data, general approaches used in the preparation of data for analysis (tokenization and normalization),

balance of the classes with methods of oversampling and under-sampling, and the employment of ML and DL algorithms. Chapter 4, “Experiment Analysis”, specifically discusses the methods used in the experiment and in the analysis of the results. It explains how to use the provided dataset to train and evaluate different ML and DL models. The findings for the assessment of the models, including precision, re-call, F1-score, accuracy, and ROC-AUC, are presented throughout the chapter using graphs, charts, and confusion matrices. The research examines accuracy-based effects of various data preparation steps as well as class balancing and feature engineering strategies on phishing detection and categorization models. Chapter 5 concludes the study contributions to the system design for detecting SMS phishing in New Zealand. It evaluates the general applicability of the work and outlines possible avenues for future research, such as investigating more complex deep-learning topologies and enlarging the applicability of the captured data.

Chapter 2 Literature of Review

A comprehensive literature review appears in this chapter by exploring how SMS phishing attacks impact New Zealand banks. This section evaluates multiple ML and DL techniques for phishing attack detection. This section performs an assessment of practical ML/DL systems to determine their usefulness against phishing attacks.

2.1 SMS PHISHING IN NEW ZEALAND BANK

The financial sector in New Zealand faces an escalating problem from SMS phishing which fraudsters call "smishing." The confidence people have in text messages allows cybercriminals to exploit this trust and obtain personal details from victims who reveal their bank information and credit card data as well as their authentication credentials. Cyber criminals craft messages that appear to be from authentic banks and urge the recipient to take some action like clicking on a link or changing information. These schemes must be able to imitate real conversation and build a sense of urgency to succeed (Mehdipour et al., 2023).

Smishing has had a major effect on New Zealand institutions and their clients. Attackers have specifically targeted well-known organisations, including NZ banks, with messages that seem to be authentic to deceive recipients. As noted in (McCombie, 2008), Due to an extensive use of mobile phones and SMS's inadequate security features, these attacks are especially successful. In the belief that they are protecting their accounts, victims frequently reply to such messages, only to become victims of financial fraud.

This has led to the formulation of countermeasures including client awareness, security technology, as authentications, as developed by banks and security specialists (Priezkalns, 2024b). However, this remains extremely challenging as the opponents switch their strategies. According to (Azeem, 2020), in particular, consumers need to learn how to avoid such tricks as phishing and receiving different kinds of unwanted texts. But there is also the need for technology applications to help prevent Spam and fraud messages amongst others—sophisticated spam filters, and fraud detection devices. Yet, as (Azeem, 2020) advises that when SMS is abused, even security measures like two-factor authentication (2FA) can be compromised.

This would suggest that future prevention of smishing is in a comprehensive system approach that factors in technological advancement, legal enforcement, and an educated public,

among others. According to (Mehdipour et al., 2023), this study reveals that there is an opportunity to enhance fraud prevention by integrating artificial intelligence into fraud detection and by enhancing collaboration among the public and private sectors. While the advancement in the sophistication of the threats in the cyberspace remains high, stakeholders need to be away of the threats and ensure that banking systems in New Zealand are protected against mobile phone phishing attacks.

(Wright et al., 2023) conducted an incident when an employee of a big university's finance department often experienced simulated email-based phishing attempts. They discovered data that backs up our theories that a person's vulnerability to phishing assaults is affected by their role in the organization's information flows and how their cognitive processing is affected by their workgroup duties. They argue that an individual's vulnerability to phishing attacks depends on factors other than their familiarity with IT security measures, such as contextualised, multidimensional effects on data processing.

(Azeem, 2020), suggests that the primary defences against SMS phishing are enhancing customer awareness. It stresses that the New Zealand banks focus on increasing customer awareness about phishing attacks. ANZ, ASB, and Westpac have adopted the use of campaigns such that websites and applications offer up-to-date information on the scams (BioCatch, 2024). This strategy has been most helpful when discouraging phishing by arming the customer with information.

(Nowitz, 2018) collected a set of the most successful emails that have passed through the company's protection system. A large university was chosen as the target organization for a simulated phishing due to its size, and the ease with which authorization for the exercise was obtained. It was easier to see and execute the sample in a virtualized environment, so no issues were mistakenly triggered by exporting scam emails into a fake account from the university's exchange server. There were two main goals in doing this step: first, to get a better understanding of the methods employed by phishers and second, to determine the most prevalent sorts of phishing emails that university employees receive.

Research on SMS phishing in New Zealand banks emphasizes on both preventive measures and customer education to combat financial fraud (Karhani et al., 2023). examined financial fraud in New Zealand, proposing a framework to help individuals avoid modern banking scams by highlighting various forms of online banking fraud and evaluating existing fraud prevention strategies. These initiatives highlight the need of continuous education and preventative actions to handle the ever-changing phishing risks in New Zealand.

2.2 MACHINE LEARNING AND DEEP LEARNING TECHNIQUES

The significance of implementing solutions based on ML and DL for better identification and prevention has grown due to the current surge in SMS scams. These technologies leverage linguistic and behavioural features to differentiate fraud messages within text communications. With the growing sophistication of cyber threats, phishing attack detection has increasingly relied on ML and DL techniques. These advanced methods offer reliable solutions for identifying phishing attempts and effectively handling fraudulent emails, URLs, and other deceptive websites.

The machine learning software has been exceptionally useful for structured data of the content of SMSs. To this end, algorithms can identify scam messages with great confidence since the models are trained on labelled datasets. Previously the traditional machine learning was chosen for this process, but deep learning is proved to be more effective for this task because it can work with unstructured data and identify patterns easily. Currently CNNs and RNNs are the models most used to analyse SMS contents. These models stand out due to their ability to extract context and semantics within messages, which is vital for phrasing out the real phishing cases from normal messages. (Salman et al., 2022) conducted an empirical analysis demonstrating how deep neural networks outperform conventional ML models in identifying scam messages.

Hybrid approaches combining ML and DL techniques have shown increased attention. (Maqsood et al., 2023) introduced an intelligent framework that integrates feature engineering and deep learning for improved SMS scam detection. The significance of using linguistic and contextual factors to respond to evolving scam tactics is emphasised by their research.

Despite the encouraging developments the system encounters obstacles during its implementation. The difference in dataset sizes that features fewer scam messages than legitimate messages create obstacles for model performance. The research by (Johnson & Khoshgoftaar, 2019), indicates that oversampling and adversarial training offer solutions to overcome these limitations.

(Prusty et al., 2022) provided a list of factors, which he observed would help one to distinguish between a genuine and an online 419 scam, after analysing the subject comprehensively. The detection of newly emerging fraud requires both additional changes to existing algorithms and new algorithm development performed manually. The method performs effective detection between scam and ham SMS messages although it necessitates increased human labour costs with expanding customer numbers and data volume. The researchers

determined that Random Forest RF classification method achieved 99.9% accuracy in their analysis.

The study in (Gadde et al., 2021) applied several approaches to DL and ML to address the task of identifying of SMS scam. The researchers constructed their scam detection model with data obtained from University of California (UCI). The empirical data shows that LSTM model establishes a 98.5% accuracy score for detecting scams surpassing the performance of existing models. All implementations occurred through python programming language.

(Amir Sjarif et al., 2019) applied the RF Algorithm and the TF-IDF method on data gathered from the sample of both real and fake SMS scam messages. The RF method demonstrates superior performance in the results of performance assessment yielding an accuracy of 97.50%.

ML applications for phishing detection require feature extraction from URL, email header and website contents to establish structured data. SVMs together with DT and RF constitute traditional ML methods which produced remarkable outcomes when processing structured data. Typical research by (Maqsood et al., 2023) demonstrated that parallel processing leads to advanced ML models which deliver high accuracy in detecting Phishing URLs. The methods have certain limitations because thorough feature engineering is often necessary while modification of new phishing patterns remains challenging.

The automated process of feature extraction and unstructured data handling through DL techniques provides better advantages than typical ML methods. The detection of phishing incidents heavily depends on the application of CNNs and RNNs models. CNNs are particularly effective in processing textual and image-based phishing data, while RNNs excel in capturing sequential patterns in phishing emails or URLs. For instance, (Alshingiti et al., 2023) highlighted the preponderance of DL methods in phishing detection studies by a comprehensive evaluation.

(Sharaff et al., 2023) created a smart model that can differentiate among smishing messages and ham communications using a combination of DL, ML, and regular expression (Regex). The goal of generating regex rules is to refine the dataset by utilizing the dataset scam messages. Some popular types of the superior models are RF, MNB, and SVM, while some of the DL models are LSTM, Bi-LSTM, layered LSTM, stacked Bi-LSTM. When comparing DL models to their ML counterparts, metrics including the rate of accurate predictions on samples set, recall ratio, precision, and F1-score are used. New fields containing regular expressions are

appended into the dataset to enhance both ML and DL models that are demonstrated to perform better than the ML models alone.

(Muzigura & Casmir, 2023) investigated the use of deception theory and discovered that most Tanzanian telecom businesses are conversant with and have been victims of social engineering. Business cooperation advantages purported erroneous remittance, sim swaps, SMS phishing, password requests, and link sharing are common forms of assaults. Potential consequences of social engineering include compromise of private information, monetary loss, harm to one's reputation, and interruptions in business operations. To prevent such attacks, companies implement security measures such as awareness programs, multifactor authentication, policies, software updates, and customer care services. Data encryption, correct email verification, SSL certificates, and phishing and malicious detection systems are some of the measures that businesses take to reduce the frequency and severity of assaults. A well-known customer care number, daily alerts, and regular security evaluations are also advised. Machine learning-based defence mechanisms are recommended against social engineering-based assaults.

(Srinivasarao & Sharaff, 2023) study proposed a classifier that combines sentiment analysis with the previously used method of classifying SMS spam. To extract the features, Word2vec Data Augmentation is utilised to a pre-processed dataset. Characteristics are then passed through six distinct feature selection algorithms, together with equilibrium optimisation (EO). The best way to categorise SMS messages is to feed the optimal components into a hybrid classifier that uses KNN and SVM. Finally, Sent WordNet and AFINN Sentiment Lexicon are used for sentiment analysis. You can see how well this framework works with these three benchmark datasets. For spam detection, the spam assassin dataset outperforms the other two with a 99.82% success rate.

(Karhani et al., 2023) developed a mixed ML model to identify phishing and smishing attempts. This model combines domain-related data taken from various sources with NLP trained on real smishing messages. Their recommendation to integrate the detection system with MIPS, an open-source platform for threat intelligence, is worth considering. Flagged phishing domains may be stored and used more effectively in this way. A model was trained and evaluated employing data that was both publicly accessible and contributed by TELUS Corp. A F1 score higher than 99% and an accuracy of 99.40% were recorded in the findings.

Hybrid frameworks that combine ML and DL approaches have also emerged, integrating the strengths of both mechanisms. (Adane et al., 2024) explored how informative feature selection enhances model performance, bridging the gap between traditional ML feature dependency and DL ability to process raw data.

(Shinde et al., 2024) employed another form of FE like TF-IDF and Principal Component Analysis (PCA) alongside the traditional ML models such as K-means, On-Negative Matrix Factorisation, and Gaussian Mixture Models. Among DL models RNN-Flatten stands alongside LSTM and Bi-LSTM which have been employed. The models demonstrated compatibility with smishing signals because they could detect linear and non-linear correlations between data points. The use of DL models happens because they identify contextual relationships and sequential patterns whereas ML models are selected due to their ability to produce performance from structured text data. The assessment of ML and DL methods depended on percentage ratio and recall as well as accuracy and precision and F-measure. The accuracy rates obtained from K-means feature extraction with vectorizer reached 91.01% for classification while achieving 94.13% accuracy in the respective test. KNN-Flatten demonstrated the highest performance.

(Mahmud et al., 2024) introduced CNN-Bi-GRU as a DL approach to jointly involve CNNs with Bi-GRUs to detect and classify smishing attacks. The raw, unstructured text data utilised in this study—the SMS Phishing Collection—had to be converted into numerical vectors before Word2Vec could be trained. Among the several detection methods tested for SMS phishing, the CNN-Bi-GRU model had the highest accuracy rate of 99.82%. The scientific study evaluates hybrid DL methods' effectiveness for SMS phishing detection to enhance mobile communication security.

(Zimba et al., 2024) proposed to find smishing dangers as they happen by combining natural language processing with ML. A developed methodology adopts NLP algorithms to analyse text messages for linguistic features thus identifying potential smishing attempts. Through real-time learning the model expands its capabilities to detect non-smishing along with smishing methods through ML algorithms. The model evaluation showed F-1 reached 0.902 while AUC reached 0.95.

(Goel et al., 2024) introduced a framework for improved content-based smishing detection that uses advanced text normalisation methods to boost detection precision. Implementation of this method enhances the capability of Naive Bayesian and other ML classifiers to recognize genuine and smishing communication types. Experimental results

demonstrate a 96.2% detection accuracy along with FPR values below 3.87% and FNR at 2.85%. The analysis used public dataset information to validate the achieved results.

(Gandhi et al., 2023) compared various methods for the detection of spam SMS. The researchers conducted analysis with three ML models that consisted of DNN, Bi-LSTM model and LSTM model. The researchers tested their proposed algorithms on SMS text messages and found strong capabilities for correct spam message detection in the experiment. The DNN delivered the highest accuracy rate of 95.6522% among the three models and Bi-LSTM achieved 93.9799% accuracy and LSTM demonstrated 92.9766% accuracy.

(Dharani et al., 2023) presented a model that integrated the TF-IDF vectorizer with the Naive Bayes approach. A dataset from Kaggle is utilized to train a model. The findings showed a 95% accuracy rate and a 100% precision rate for the model.

(Abayomi-Alli et al., 2022) analysed the Ex AIS SMS-dataset and compares the outcomes of several learning algorithms. The Bi LSTM model performed at an average level of 93.4% accuracy and 98.6% accuracy when handling UCI datasets while exceeding the performance of alternative ML classifiers. The model reached higher accuracy rates than NB, Bayes Net, SOM, DT, C4.5, and J48. The proposed Bi LSTM model achieved superior results for SMS spam message classification through different metrics like accuracy, precision, re-call, and F-measure when compared against traditional ML classification methods.

(Mambina et al., 2022) utilized a model based on ML to categorise Swahili Smishing texts sent to people using mobile to phish for money. The study confirmed that TEC feature selection and RF with TFIDF vectorisation produced the most optimal model which achieved 99.86% accuracy. The Multinomial Naïve Bayes (MNB) baseline model serves to measure the produced results. The research team examined their model against classic classifier models as well. The model reached a Log-Loss value of 0.04 along with minimal FP values of 2 and minimum FN values of 4. The performance evaluation relies on a Swahili message collection with 32259 entries.

(Jain et al., 2022) incorporated ML algorithms and NLP methods. Datasets salvaged from the UCI ML Repository were subjected to four primary classification models. Since SVM achieved a rate of 98.797 percent in the accuracy examination, it was deemed the premier model.

(Aliza et al., 2022) purposed to identify spam communications to forestall various forms of cybercrime, since these messages pose a real threat to online safety in the modern era. The

accuracy of this system improves through methods that include SVM, K-NN, NB, RF and LR alongside other techniques that address SMS spam. The 99% accuracy rate achieved by SVM indicates its strength as a potential advanced ML system for future research and application.

The improvements do not solve all problems since big datasets need human labels while imbalance within phishing datasets can skew model training results. The methods of synthetic data creation together with adversarial training approaches have been proposed to handle these problems (Mughaid et al., 2022).

(Rifat et al., 2022) research demonstrated a novel approach to real-time spam SMS categorisation by using a pre-trained Google BERT model for universal spam detection. For performance evaluation, F1 scored 0.97, and the overall accuracy approached 99%.

(Ghourabi, 2021) presented "SM-Detector," a hybrid security model designed to identify smishing messages sent through mobile devices. To make "SM-Detector" more effective, they integrated three distinct detection techniques: (i) recognising harmful URLs; (ii) using regular expression analysis to identify words, phone numbers, and emails that may be spam; and (iii) sorting communications into spam and valid categories using algorithms based on BERT. They also developed an SMS monitoring and reporting app as part of "SM-Detector" system for those who prefer to use their mobile devices. This system can handle communications with both Arabic and English languages with a 99.63% accuracy.

(Ulfath et al., 2021) presented an AI system capable of automatically detecting phishing SMS by extracting robust characteristics from the messages. This model gathers superior characteristics from various ML and DL systems through its hybrid architectural design. The Phishing Detection system implements supervised CNNs together with Bi-directional GRUs and uses a pretrained transformer model MPNet. The system seeks to detect complex pattern-based unstructured short phishing messages effectively.

(Ora, 2020) strived for effective and low-latency spam message classification. Researchers utilized XGBoost, Light GBM and BNB as ML models in their study because these models demonstrate fast performance and low time complexity. Message length became an additional characteristic to Unigram and Bigram within the TF-IDF matrix extraction process. Chi Square feature selection helped decrease space complexity after its implementation. The findings showed that Light GBM and TF IDF matrix-based BNB were the most successful techniques. Light GBM ran in 0.157 seconds to deliver 96.5% accuracy and reached 95.4% accuracy in 1.708 seconds.

(Rashid et al., 2020) showed an effective method for detecting phishing attempts based on SVM classifier. Tests have demonstrated that the developed method successfully differentiates safe from dangerous websites by achieving a 95.66% accuracy rate from a single 22.5% reduction in the unique capability usage. Results obtained using "UCI" standard phishing datasets in benchmarking tests show that the proposed method shows promise.

In conclusion, the use of ML and DL in identifying SMS scams has significantly advanced the fight against cybercrime. Through their implementation organizations gain the ability to secure consumers by enhancing detection accuracy while performing real-time monitoring. Multiple ML and DL approaches have been analysed in research for detecting harmful messages, but they differ in performance capabilities and scalability aspects. Table 1 summarizes several studies that utilize ML and DL techniques for analysing and detecting SMS scams, highlighting the models used, datasets, accuracy, and key insights into SMS phishing (smishing) detection.

Author(s) - Year	Aim	Technique	Finding
Salman, Ikram, and Kaafar (2022)	To evaluate various ML and DL models on a large SMS scam dataset and assess adversarial resistance.	Deep Neural Networks (DNNs), semantic and syntactic feature models	DNNs outperformed conventional ML; highlighted the importance of adversarial robustness.
Prusty et al. (2022)	To establish criteria for scam detection and improve classification methods.	RF classification method, 10 scam detection criteria	Achieved 99.9% accuracy; required manual adjustments for evolving scams.
Gadde, Lakshmana Rao, and Satyanarayana (2021)	To implement and compare various ML and DL strategies for SMS scam detection.	LSTM, other DL models	LSTM achieved 98.5% accuracy, surpassing other models.
Amir Sjarif et al. (2019)	To process SMS scam messages using advanced techniques.	RF algorithm with TF-IDF	RF achieved 97.50% accuracy, outperforming other algorithms.
(Maqsood et al., 2023)	ML with parallel processing for phishing detection	Decision Trees, Random Forests, SVMs	Demonstrated the utility of parallel ML processing for phishing detection with high accuracy.

Adane, Beyene, and Abebe, 2024	Hybrid ML/DL with feature selection for phishing detection	TF-IDF, PCA, RNN, LSTM, Bi-LSTM	Informative feature selection bridges ML and DL performance gaps.
Mahmud et al., 2024	CNN-Bi-GRU for SMS phishing detection	CNN, Bi-GRU, Word2Vec	Hybrid model excels at SMS phishing detection with 99.82% accuracy.
Zimba et al., 2024	NLP-based real-time smishing detection	NLP, ML (unspecified algorithms)	Achieved F1-score of 0.902 and AUC of 0.95 for real-time detection.
Gandhi et al., 2023	Comparative analysis of DL models for SMS spam	DNN, Bi-LSTM, LSTM	DNN outperformed Bi-LSTM and LSTM with 95.65% accuracy.
Abayomi-Alli, Misra, and Abayomi-Alli, 2022	ML and DL comparative analysis on SMS datasets	Bi-LSTM, NB, SOM, DT, J48	Bi-LSTM showed significant improvements (93.4%-98.6%) over traditional ML classifiers.
Ora, 2020	Lightweight ML models for SMS spam detection	XGBoost, LightGBM, Bernoulli Naïve Bayes	Bernoulli NB achieved 96.5%, LightGBM 95.4% with low-latency detection.
Ghourabi, 2021	Hybrid model (SM-Detector) for phishing detection	BERT, regex-based methods	Combined malicious URL, regex analysis, and BERT for 99.63% accuracy.
Mambina, Ndibwile, and Michael, 2022	ML with feature selection for Swahili SMS phishing	Extra Trees, Random Forest, Multinomial Naive Bayes	Focused on Swahili SMS phishing, achieving 99.86% accuracy with the lowest false positives/negatives.
Sharaff, Pathak, and Paul, 2023	Hybrid ML/DL with regex rules for smishing detection	RF, MNB, SVM, LSTM, Bi-LSTM, stacked Bi-LSTM	Regex rules enhanced ML/DL models, improving precision, recall, and F1 score.
Muzigura and Casmir, 2023	Defence mechanisms against social engineering	ML-based defence mechanisms	Highlighted prevention measures for social engineering attacks.
Srinivas Rao and Sharaff, 2023	Hybrid ML/DL with optimization for spam detection	KNN-SVM hybrid, RSO optimization, AFINN, Sent Word Net	Achieved 99.82% accuracy on Spam Assassin dataset using equilibrium optimization.
Ulfath et al., 2021	Hybrid DL for phishing SMS detection	MP Net, CNN, Bi-GRU	Hybrid approach outperformed individual ML/DL models.
(Karhani et al., 2023)	Hybrid ML/NLP for phishing detection	NLP, phishing domain features	Achieved 99.40% accuracy with F1-score > 99%, integrating phishing detection with MISP.
Goel et al., 2024	Content-based smishing detection	Naive Bayes	Achieved 96.2% accuracy with FPR of 3.87% and FNR of 2.85% through advanced text normalization.

Table 1: Critical comparisons of existing models on ML/DL in SMS phishing detection

A previous study is important as they show the advantages of integrating ML and DL algorithms for SMS fraud detection. Using these technologies, it is possible to analyse huge amount of information, and to accurately determine complex tendencies in the language and behaviour. Using structured datasets, a performance metrics of ML models such as RF and SVM are strong and stable. However, DL models such as LSTM networks and CNNs are far better suited to processing unstructured data and to derive deep complex features from the textual info providing better context and semantic analysis. Moreover, an integration of ML and DL techniques of a hybrid methods improves a detection to capacity and flexibility in responding to new emerging scams. It also brings real time analysis, where organizations can detect and prevent fraud, making these technologies proactive measures to prevent digital fraud.

2.3 CRITICAL COMPARISONS OF EXISTING MODELS

Traditional ML models have been outperformed by DNNs in SMS phishing detection. However, their main limitation lies in their reliance on continual updates and manual interventions to adapt to evolving scam patterns. This makes DNNs less adaptable over time. The Random Forest (RF) has achieved near-perfect accuracy with scam detection criteria. Despite its strong performance, RF models require manual rule adjustments, which limits their scalability and effectiveness in real-time scenarios, especially when faced with new or evolving scam tactics. Phishing detection benefits from the LSTM models' exceptional ability to capture long-term dependencies in text. To function at their best, LSTMs need huge, high-quality datasets and are computationally demanding. This can be a drawback in dynamic or resource-constrained environments. The Hybrid models combining ML and DL techniques, such as Random Forest with Bi-LSTM, show improved accuracy. Their high computing resource requirements and the added complexity they bring to model adjustment may prevent their widespread use in the real world.

2.4 RESEARCH GAPS

Several research gaps remain in the domain of SMS scam detection using ML and DL. One limitation is the absence of a more merged set of benchmark datasets, which sometimes becomes a problem when comparing results from one model to another. The datasets that are available for the public do not contain numerous amounts of knowledge or expand across large ranges of scam messages as observed in various actual-world presences. The inadequate investigation of model's resistance to hostile attacks and their capacity to generalise to new cases represents another significant gap. Furthermore, although literature research has shown novelty of the hybrid approaches, studying their cooperation between ML and DL to handle the

issues of imbalanced datasets and dynamic changes in the scams is still inadequate. The future studies should pay attention to the accumulation of more comprehensive dataset, increasing the resistance of model to adversarial risk and the establishment of large-scale solution with relatively high accuracy and lower computational complexity

2.5 PHISHING TRENDS IN NEW ZEALAND COMPARED TO GLOBAL TRENDS

Phishing attacks in New Zealand often have a localised focus, using cultural references and exploiting trusted local institutions like government agencies or banks. This makes them more relevant and deceptive to the local population.

While global phishing trends are diverse, targeting various platforms with sophisticated methods, New Zealand experiences a high prevalence of SMS phishing, which is more common due to the widespread use of mobile devices and SMS communication.

Cybersecurity reports from New Zealand show an increase in smishing campaigns, with localised targeting strategies. This requires phishing detection models to focus more on mobile-targeted threats, in contrast to global phishing attacks that often involve broader social engineering or malware tactics.

New Zealand's unique phishing landscape calls for tailored detection systems that account for the specific targeting methods and characteristics of local attacks. Incorporating insights from local cybersecurity reports will improve the effectiveness of detection systems in the country.

2.6 TOOLS AND TECHNIQUES

This study used a mix of ML, DL, and data preparation methods to identify SMS smishing. The dataset was cleaned and preprocessed using Pandas, NLTK, and WordNet lemmatization, followed by tokenization and word embedding with Word2Vec. The SMOTE approach was used to remedy the imbalance in classes. Features were extracted and encoded before the dataset was split into training and testing sets. The ML models we created using Scikit-learn, like XGBoost and Random Forest, produced very accurate results. TensorFlow and Keras were utilized to construct DL models, with CNN demonstrating superior performance compared to RNN and LSTM. Accuracy, precision, re-call, and F1-score were the measures used for performance assessment.

2.6.1 Text Tokenization

The process of text embedding involves converting the text into a meaningful vector of numbers after creating an encoded representation of the text. When working with textual

material, this operation is crucial for ML algorithm analysis (Hapase & Patil, 2024). The need for numerical input in ML models makes tokenisation a crucial step in text processing. Deconstructing text into its component parts, often words or phrases, is called tokenisation (Valerian Chinedum et al., 2023).

2.6.2 Word2Vec Embedding

The data is prepared to be vectorized to generate word embeddings after tokenization. This effort included the construction of word embeddings, with training and test sets of embeddings being developed independently (Jáñez-Martino et al., 2023). As a result, it will return a row of NaNs if a word in the test set is not in the embedding vocabulary. Results showed that these conditions had no effect on performance when using the suggested re-engineering process (Mikolov et al., 2013). An average of a word vectors is calculated for each SMS message to create a fixed-size vector representation for each message. For example, a phishing SMS like "Your bank account has been locked due to suspicious activity. Verify your details at <http://fake-bank.com>" is first tokenized into words. Each word is then mapped to a pre-trained Word2Vec vector, while unknown words (e.g., phishing URLs) may return NaN values

2.6.3 Data Balancing Using SMOTE

SMOTE (Xu et al., 2019) creates synthetic minority class samples to increase minority class samples in datasets with a very uneven ratio. To avoid the overfitting issue, we generated fresh samples synthetically, which differed from the multiplication method. Using interpolation methods between nearby samples, SMOTE creates data samples for the under-represented group (D.-C. Li et al., 2022). Therefore, SMOTE enhances the classifier's generalisability by enhancing the quantity of samples from minority classes in an imbalanced dataset. The following is an explanation of the formal SMOTE procedure: The initial step is to decide on the preferred quantity of oversampling, N , as an integer. This figure can be chosen since it balances the dataset with a 1:1 ratio between the various classes. Iteratively, three primary steps should then be followed. The rules of the procedure are as follows: 1. Choose a sample at random by a minority group; 2. Determine the sample's K (by default, 5) closest neighbours; and 3: Randomly select for interpolation, N of these K neighbours are to generate additional samples (D. C. Li et al., 2022). A basic knowledge of how SMOTE functions is shown in Figure 1. To maintain parity between the classes, N is the number of replicates of each minority group that must be produced (e.g., a 1:1 ratio). K refers to the number of nearest neighbours considered when creating new synthetic samples. For example, if the dataset has 100 phishing (minority class) and 1,000 legitimate (majority class) messages, setting $N = 10$ means generating 10

synthetic phishing messages per existing phishing sample. If $K = 5$, the algorithm selects five closest phishing messages and interpolates between them to generate new samples, increasing dataset diversity and balance.

Synthetic Minority Oversampling Technique

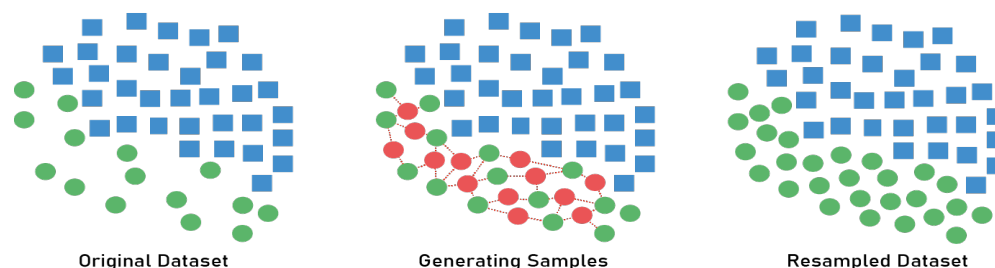


Figure 1. An illustration of SMOTE (Synthetic Minority Over-sampling Technique) applied to a classification scenario that does not achieve balance. SMOTE creates new instances of the minority class by interpolating between nearby samples (Jefferson, 2020).

2.6.4 XGBoost-Classifier-Model

For gradient boosting, XGBoost is an enhanced distributed library that proficiently executes gradient-boosting algorithms (T. Chen & He, 2014). This tree-based methodology incrementally trains several decision tree models and combines them to improve predictive accuracy. Using the gradient boosting technique, XGBoost creates a new model in each iteration, progressively enhancing the prior model's performance. In every iteration, the latest model will fit the previous model's residuals. The decision tree is the core learner of XGBoost. A classification and regression tree (CART) is commonly employed as the foundational DT. However, other varieties of tree models are also allowed. To minimize the loss function, nodes are continuously split to create each decision tree gradually (Gualberto et al., 2020). To limit an intricacy of a model and avoid overfitting, a regularisation term is incorporated. Limiting the node splitting and pruning the tree's leaf nodes are used to prevent too complicated models. Every feature in the model can have its importance score provided by XGBoost. It optimizes calculation time and memory while supporting Python and other languages. This research employed statistical acoustic characteristics as feature vectors for the XGBoost model. Upon the calculation of each feature, it is combined into a feature vector and appended to the feature list, which is designated for the training and testing of ML models. Standardizing feature vectors improves model training and performance (Al-Sarem et al., 2021). Figure 3 displayed a general working of the XGBoost classifier.

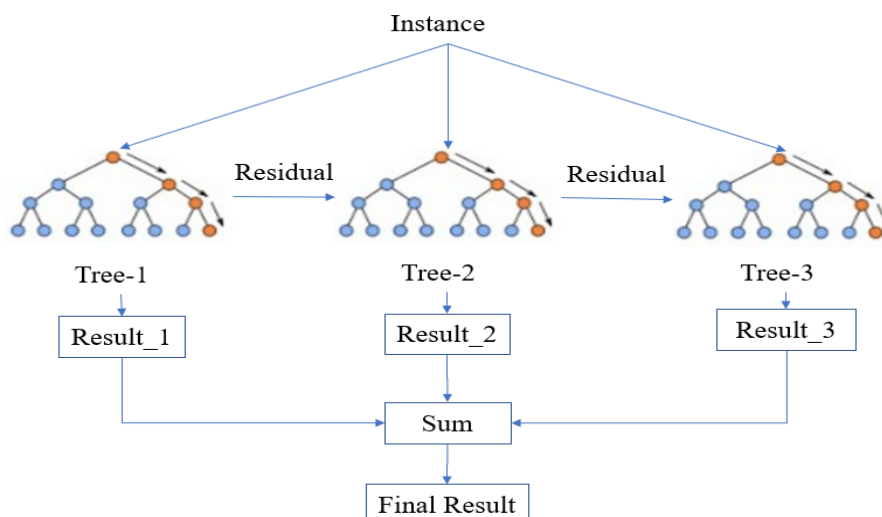


Figure 2. General architecture of XG boost classifier(T. Chen & Guestrin, 2016)

Figure 2 illustrates the general architecture of the XGBoost classifier, a powerful ensemble learning method. It depicts a sequential process where an input "Instance" is passed through multiple DT (Tree-1, Tree-2, Tree-3). Each tree, represented as a branching structure with nodes and leaves, makes a prediction based on the instance's features. The first tree (Tree-1) generates an initial prediction (Result_1), and an error or "Residual" among this prediction and an actual value is calculated. This residual becomes the target for the next tree (Tree-2), which aims to correct the errors of the previous tree, producing Result_2. This process repeats for subsequent trees (Tree-3), each focusing on minimizing the residuals left by the preceding trees. Finally, the predictions from all trees (Result_1, Result_2, Result_3) are summed to generate the "Final Result," which represents the aggregated and refined prediction of the XGBoost model. This iterative, error-correcting approach is the core of XGBoost effectiveness.

2.6.5 Random-Forest-Classifer-Model

RF classifies data by employing many DT that are based on various components of the information. It integrates these trees' predictions to enhance model accuracy. The classification scheme in this study integrates multiple features from structured data (Alnemari & Alshammari, 2023). A goal of using the RF classifier is to identify patterns in unobserved variables for better prediction. The classifier known as the random forest develops a model based on trees that incorporates unified information, predicts numerous class dimensions, and stresses just the most critical elements relevant to the classification process. This method offers the benefit of highlighting just significant features and achieving a greater rate of accuracy than other simple categorization methods(J. Chen et al., 2020). RF classifiers can be employed to develop an

extension's categorisation model that predicts and categorizes user perceptions of a new technology's usability.

There are two separate steps to the Random Forest algorithm's operation. The early stage involves the construction of the RF by aggregating N DT. A subsequent phase entails generating predictions based on every decision tree developed during the first phase(Zaini et al., 2020). Figure 3 displayed a working of RF model.

A process is established by the steps and diagram below:

- **Step 1:** Choose K training data points randomly means that, instead of using the entire dataset to train every DT in the RF, a subset of the data is selected randomly.
- **Step 2:** Generate DT for a chosen datapoints in step two.
- **Step 3:** Create N decision trees in step three.
- **Step 4:** Go through Steps 1 and 2 again.
- **Step 5:** After each decision tree has predicted a value, sort the new data points into the most popular category.

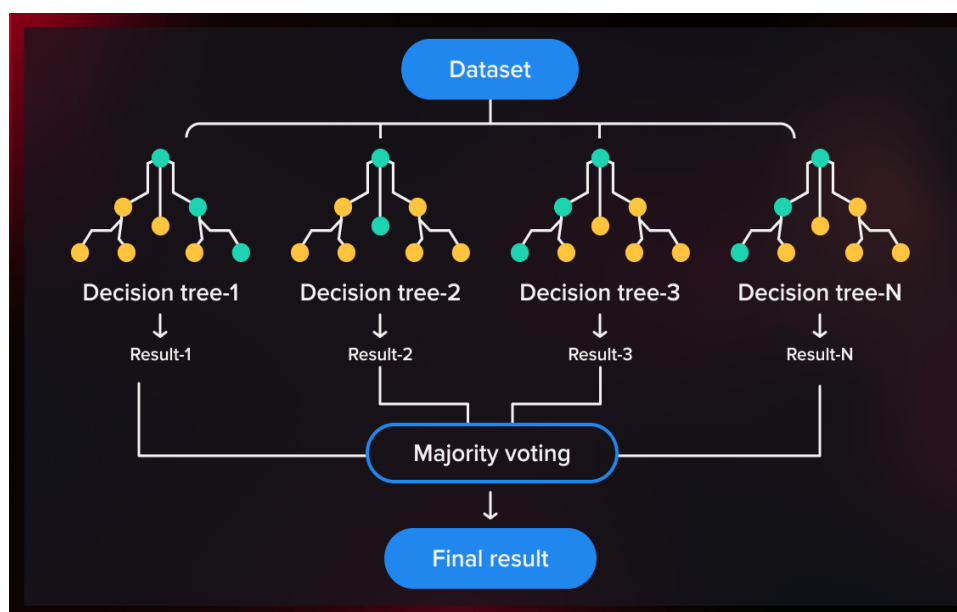


Figure 3. Working of random forest classifier (Logunova, 2022)

2.6.6 Convolutional-Neural-Networks (CNN) Model

The need for manual feature extraction is eliminated by the discriminative DL architecture called a CNN, which can learn directly from input data. NLP, picture segmentation, medical image analysis, visual recognition, and other applications benefit from this network's ability to handle a variety of 2D shapes. It is more successful than a conventional network since it automatically identifies crucial input items without human intervention (Jacovi et al., 2018).

i. Convolutional Layer

A key component of CNNs, the convolutional layer applies multiple filters (or kernels) to an input data to extract important features. Each kernel has a specific width and height, and its weights are initially assigned randomly. During training, these weights adjust to improve feature detection. For phishing detection, the CNN applies filters to SMS text data, learning to recognize patterns like suspicious keywords or URL structures. Feature maps are the representational ends of this process, which depict the features collected from an input data. Working in a high-dimensional feature space, the kernel reduces calculations, allowing the model to discover complicated phishing patterns. CNNs can identify complex phishing attempts because the kernel approach converts a basic linear model into a more effective non-linear one (Nagy et al., 2023).

ii. Pooling

A pooling layer, which is also called the down-sampling layer, streamlines the feature map by reducing its dimensionality without losing any important data. A filter slides across incoming data in the pooling layer (max, min, and average) to pool it.

Through down-sampling, pooling decreases a complexity of upper layers. It may appear to be a reduction in resolution in image processing. The number of filters is not impacted by pooling. One popular pooling method is max pooling. Maximum values in the dataset rectangular subregions are given back. 2x2 is the most common max-pooling size. (Zafar et al., 2022). The basic structure of the CNN Model is shown in Figure 4.

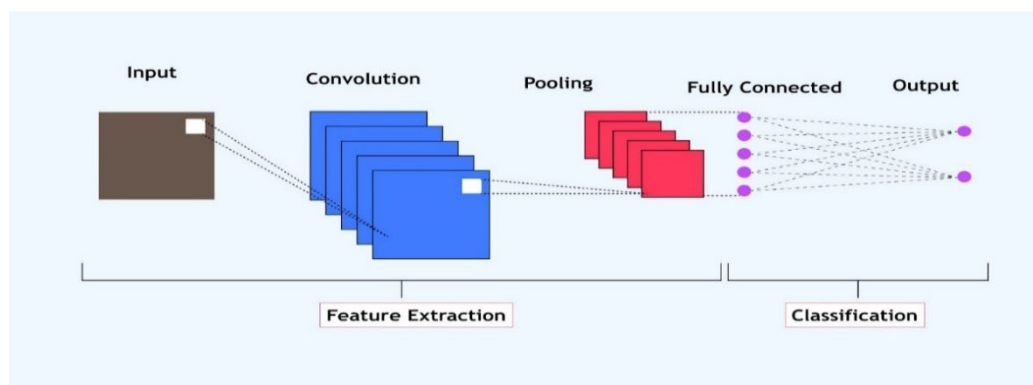


Figure 4. Fundamental architecture of the CNN model (Do et al., 2021)

Feature of CNNs: The way the model's weights are distributed ensures that small shifts in the input don't affect its performance. The learning function's filters can be useful in any setting by identifying important features. The ability to identify patterns depends on how well the filters perform after being trained with randomised data. However, when considering the

importance of specific locations in an input, using the same weight everywhere may not always be the best approach.(Mohammed et al., 2022).

iii. Activation Function

Non-linear activation functions like ReLU help the model capture complex relationships in phishing messages. It allows CNN to differentiate among legitimate and phishing messages by emphasizing critical features while ignoring irrelevant ones.

iv. Fully Connected Layer

This layer combines extracted features and makes the final classification decision. It takes the processed information and assigns a probability score to determine whether an SMS message is "ham" (legitimate) or "smish" (phishing).

2.6.7 Recurrent-Neural-Networks (RNN) Model

RNNs are networks with directed connections between their nodes. The network ability to detect sequential dependency on data is hidden in its state (memory), which is its essential feature. Because GRU, it is a type of RNN architecture introduced as an alternative to LSTM networks, struggle to retain long sequences of data, we opted to employ LSTM networks instead. By omitting the convolution layer and substituting the three-gated LSTM layer—which is identical to the CNN—in the RNN design, we achieve the following results: The input gate's control modifies the hidden state, the forget gate's control decides what data gets removed from memory, and the output gate's control decides how memory is output. The output by a densely linked layer is constructed using the output from this layer(Murali & Swapna, 2019). Figure 5 shows a working of RNN model.

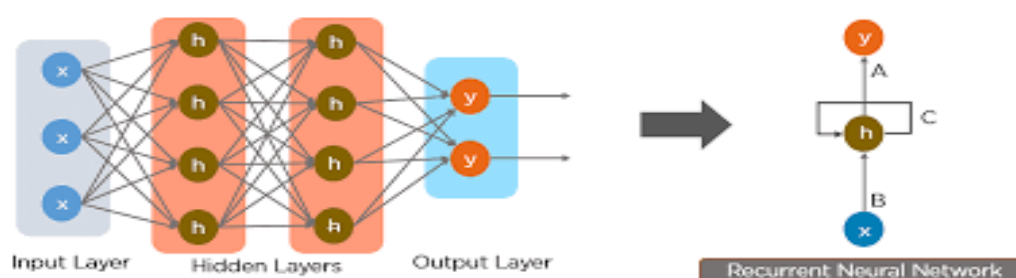


Figure 5. Working of RNN Model (Gupta et al., 2024)

2.6.8 Long-Short-Term-Memory (LSTM) Model

A LSTM network is an example of a RNN. Its design is defined by the presence of at least one cycle of connections between neurones. It was Hoch Reiter and Schmid Huber (Hochreiter & Schmidhuber, 1997)who first proposed it and were improved upon in the years that followed. Long-term dependencies are specifically captured by LSTM networks, may be able to overcome

the vanishing and exploding gradients that were previously inherent to RNNs. Figure 6 shows the LSTM memory block's architecture.

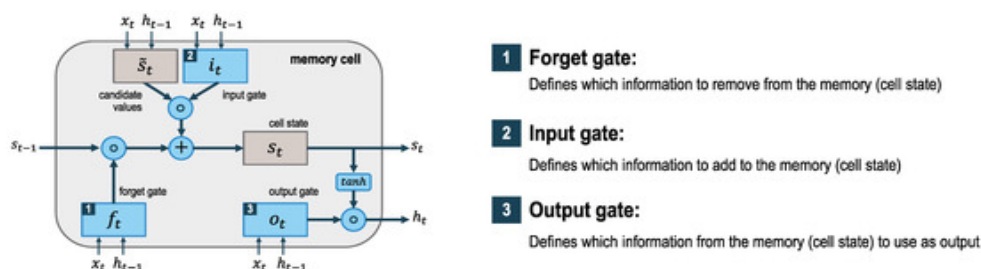


Figure 6. The LSTM memory block's structure (Su, 2020).

An LSTM network has an input layer, a memory cell (or cells), and an output layer. Each explanatory variable has an identical number of neurons in the input layer. The so-called memory cells that make up the buried layer of an LSTM network are its defining feature. Each memory cell contains three gates: the forget (f_t), output (o_t), and input (i_t) gates. Maintaining and altering a cell state (s_t) is the responsibility of these gates.

At time-step t , each of the 3 gates receives one element of the input x_t and the memory cell output h_{t-1} by a previous time-step $t-1$. The gates are filters with specific functions: The forget gate chooses which cell data to erase. The info added to the cell state is controlled by the input gate. An output gate chose which cell state data to use (Fischer & Krauss, 2018).

2.6.9 Simulation Tool

This research on SMS smishing detection was conducted employing Python in Jupyter Notebook, leveraging various libraries for data preprocessing, ML, and DL.

1) Python

The project selected Python as its main programming language because it has rich collection of DL modules and NLP and ML libraries. This system made it possible for data handling, along with feature extraction and model training, followed by evaluation operations without interruptions (Mehare et al., 2023).

2) Google Collab

Google Collab was used as an alternative environment, offering **free GPU and TPU support** to accelerate deep learning model training. It provided advantages such as (Prabanjan Raja, 2022):

- Cloud-based execution (no local setup required)
- Integration with Google Drive for dataset storage
- Support for TensorFlow and Keras with high computational power

3) Jupyter Notebook

Jupyter Notebook served as the development environment through which users could efficiently operate Python code using its interactive interface. The application enabled programmers to execute each phase of data development, including pre-processing work as well as visualization tasks and model-building steps (Källén et al., 2021).

4) Pandas

The data processing required Pandas as its primary tool for manipulating and pre-processing the data. Pandas read data from a CSV file while carrying out operations for handling missing values and removing duplicates as well as message filtering. The tool enabled analysis of the dataset and calculated statistics as well as drawing distribution visualizations of classes (Snehkunj et al., 2022).

5) NLTK (Natural Language Toolkit)

Tokenisation, stop word elimination, and lemmatisation were some of the text preparation tasks for which NLTK was used. The word tokenize function helped split SMS messages into individual words, while lemmatization using the WordNet Lemmatizer reduced words to their base form, improving text standardization (Bird et al., 2013).

6) Scikit-learn

Several ML tasks made use of Scikit-learn (Pedregosa et al., 2011):

- **Text Vectorization:** A dataset was transformed into numerical representations using Word2Vec.
- **Data Splitting:** The dataset was divided into two parts: one for testing and one for training using the train test split method.
- **Machine Learning Models:** Classifiers like XGBoost and RF were implemented.
- **Data Balancing:** The SMOTE from Scikit-learn was utilized to balance a dataset.
- **Evaluation Metrics:** Precision, re-call, F1score, and accuracy were calculated using classification report.

7) TensorFlow and Kera's

DL models were built using TensorFlow and Kera's. Different architectures were implemented:

- **CNN (Convolutional Neural Network):** Used for pattern recognition in SMS text data.

- **RNN (Recurrent Neural Network):** Created with the purpose of capturing textual sequential dependencies.
- **LSTM (Long Short-Term Memory Network):** A specialised RNN trained on sequential data with an emphasis on long-term dependency.

Binary cross-entropy loss in conjunction with the Adam optimiser was applied for evaluating different models by testing them against sample data. The model performance assessment included both accuracy curves with classification reports.

Chapter 3 Research Methodology

The proposed research technique with its data preparation and preprocessing as well as text representation method receives clarification in this chapter. A description of the proposed effective classification method that uses both ML and DL models follows.

The research targets designing a domain-specific SMS phish detection system utilizing both the Kaggle SMS Smishing Collection and authentic examples obtained from New Zealand DIA. A goal exists to design an efficient model which detects SMS phishing messages using ML and DL technologies while resolving class distribution and data ethics for better cybersecurity practices. The research investigates and evaluates various ML and DL models before determining which one most effectively satisfies the New Zealand market requirements. The dataset consisting of SMS Smishing Collection from Kaggle and smishing messages from the New Zealand DIA anti-scam archive fulfills the requirement of local context relative to New Zealand. The Pre-processing methods involved steps to manage missing and duplicated values while checking label uniqueness and performing text pre-processing and lemmatization followed by label encoding. The Tokenization is used to extract features and apply Word2Vec embeddings for vectorizing messages, while SMOTE balances the class distribution. The data is then divided into 80% for training and 20% for testing. The next step is the implementation of ML and DL classification models, which may include RF and XGBoost, CNN, RNN, or LSTM. We compared the models' precision and recall, finding the one that was most effective in smishing message detection. evaluations of each other's performance based on the F1score and accuracy. All experiments are run using an Intel Core i7 processor, 16GB RAM, and an NVIDIA RTX GPU, running Python with TensorFlow and Scikit-learn.

Figure 7 shows a proposed block diagram that shows the overall steps of model implementation.

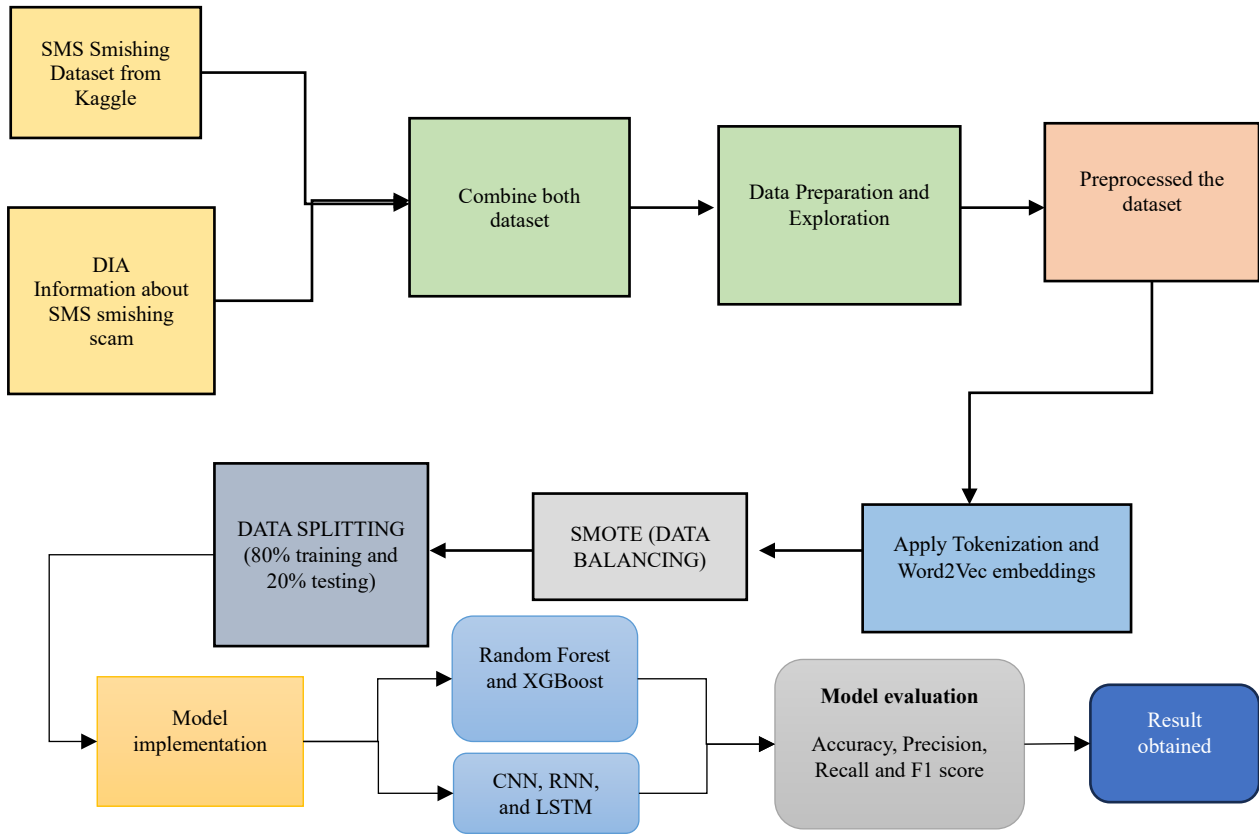


Figure 7. The Proposed Block diagram of the methodology to build and evaluate the ML/DL detection system for smishing

3.1 DATA PREPARATION AND EXPLORATION

The dataset for detecting SMS smishing messages consists of 5,884 entries with two columns: "Label" and "Text." The "Label" column categorizes messages as either "ham" (legitimate) or "smish" (phishing). The "Text" column contains the SMS message content. This dataset combines the SMS Smishing Collection from Kaggle and smishing examples from the New Zealand DIA anti-scam archive (Priezkalns, 2024a), ensuring it is relevant to the local context. It includes key attributes such as message content, sender and receiver details, timestamp, message type, and URLs. This dataset goes beyond simple SMS classification by incorporating metadata like sender and receiver details, timestamps, and message types. These elements allow for a detailed analysis of smishing patterns, enabling ML models to detect malicious messages based on both text content and behavioural indicators (e.g., suspicious URLs, message timing, sender trends). The research capitalizes on verified examples sourced from New Zealand DIA's anti-scam database to connect theory to practical phishing scenarios users encounter.

Due to the research's focus on the New Zealand market, it makes use of both the Kaggle SMS Smishing Collection and the New Zealand DIA anti-scam collection to compile real

phishing messages. Adding local smishing messages from New Zealand user data to a publicly available dataset improves detection accuracy while enhancing a model's ability to generalize for New Zealand users.

3.1.1 Loading the Dataset

The SMS smishing detection analysis retrieves dataset information by a CSV file through panda library. The dataset contains two labelled columns namely "Text" which displays SMS content and "Label" which indicates whether the SMS is genuine or a smishing attempt.

3.1.2 Extracting Short Messages:

The researchers apply selection criteria to their database which identifies the "ham" (regular) and "smish" (phishing) messages. The selected SMS messages from both categories can be found by sorting messages by their length followed by displaying the ten shortest ones from the "ham" and "smish" classes. Short text classification becomes more robust after this step because it trains the model to handle brief texts correctly and avoid context-based errors. The system design matches actual smishing scenarios because attackers tend to deploy short messaging formats. The process becomes a part of pre-processing and data exploration work that supports both feature extraction and model enhancement to improve phishing detection efficiency.

3.1.3 Class Distribution

The distribution analysis of "Label" column establishes the proportion among "ham" and "smish" messages. The detection of class imbalance problems at this phase becomes critical because it can negatively impact machine learning algorithms when left unaddressed. The dataset's class distribution is based on the percentage of "ham" (legitimate) communications to "smish" (phishing) messages. Machine learning models may be skewed in their predictions if there is an imbalance in the distribution of classes, such as when one class is noticeably larger than the other. Class distribution analysis is not part of pre-processing itself but informs pre-processing decisions. If an imbalance is found, pre-processing may include class balancing techniques (e.g., SMOTE).

3.1.4 Pre-processing

Raw data from SMS messages must first undergo pre-processing before being fed into the DL model. It is necessary to clean the input text so that it is free of unnecessary information. Pre-processing is crucial for cleaning and structuring text data before model training. Steps include handling missing and duplicate values to maintain data integrity, checking label uniqueness to prevent mislabelling, and performing text pre-processing and lemmatization to

standardize words. Label encoding converts categorical labels ("ham" and "smish") into numerical values, making them suitable for ML algorithms.

- **Handling Missing and Duplicate Values:** A thorough check of the dataset determines null value frequencies alongside the removal of duplicates. The pre-processing ensures the dataset maintains purity and reliability for investigative purposes and modelling needs. The removal of duplicate entries along with controlling missing value handling enables us to generate unbiased results and improved data consistency.
- **Unique Labels:** The unique values in the "Label" column are printed to verify that there are only two distinct categories: The database contains "ham" categories for valid correspondence together with "smish" categories for fraudulent messages. Such verification maintains a well-defined consistent structure for the classification task.
- **Text Pre-processing (Cleaning):** Pre-processing occurs on the data found in the "Text" column before noise elimination preserves significant content. The text pre-processing starts with converting words to lowercase then follows with URL and email replacement with placeholders and deletion of numerical data together with punctuation symbols and stop words along with consolidating multiple spaces. A cleaning process transforms the data into a state that enables better extractable features within suitable models.
- **Lemmatization:** A lemmatization's primary goal is to normalize words from their different forms back to fundamental bases. Lemmatization accomplishes base word normalization by applying vocabulary and morphological methods to strip word endings which yields the dictionary-form lemma. During lemmatization the system would determine between see or saw as the lemma based on the exact word usage. Model efficiency and accuracy improve through lemmatization by considering different word forms as equivalent representations of each other.
- **Label Encoding:** The "Label" column transitions to numeric values ready for machine learning model applications. A conversion assigns the binary codes 0 to ham and 1 to smash while maintaining the target variable format needed by classification models.

Before the pre-processing phase, the dataset contains raw SMS messages with potential inconsistencies, like duplicate entries, missing values, and unstructured text with URLs, punctuation, stop words, and varying letter cases. The "Label" column categorizes messages

into "ham" (legitimate) and "smish" (phishing), but in its initial form, the dataset may include inconsistencies that could affect model training. The discussed pre-processing steps transform the dataset into a structured format which makes it cleaner. The removal process gets rid of duplicates and gaps to protect the consistent integrity of data. The text content receives lowercase conversion while removing elements including URLs, emails, numbers along with punctuation to achieve message uniformity. Through lemmatization the text receives additional refinement by standardizing words to their fundamental forms to obtain word variation consistency. As a last step label encoding enables the conversion of categorical values into numbers which support machine learning algorithms. The transformation process improves data quality and supports better feature extraction which results in more reliable SMS smishing detection mechanisms.

3.2 FEATURE EXTRACTION AND EMBEDDINGS

Tokenisation for SMS text processing requires the conversion of extensive raw texts into abbreviated elements known as words or tokens. Users of Python language can tokenize their text data when it utilizes the NLTK library. Tokenise () processes a text input to generate a list of words that results from splitting text into distinct parts. Middle and lower frequency words serve as tokens and one can adjust token ways to fit their data collection and requirements. Text data automatically subdivides into tokens using whitespace with punctuation and single-letter tokens. Addition of rules for tokenisation can be achieved using regular expressions and other methods to design custom rules for specific requirements.

To convert text messages into numerical representations, tokenization is applied to split messages into meaningful units. Afterwards, the semantic links between words are captured by transforming words into dense vector representations using Word2Vec embeddings. As a result, the algorithm can identify phishing trends more accurately by using vector representations of words with similar meanings.

3.3 CLASS BALANCING WITH SMOTE

Class imbalance may cause biased predictions since phishing mails are less common than real ones. We use the SMOTE to generate synthetic instances of the minority class to ensure that the dataset is balanced. As a result, the model can identify smishing messages more correctly and avoid favouring the majority class ("ham").

To evaluate an impact of SMOTE, a comparative performance analysis was conducted by training the models both with and without the application of SMOTE for class balancing. The findings showed that SMOTE effectively addressed class imbalance by significantly improving

re-call and F1score, especially for the minority class. Without SMOTE, models like RF and XGBoost exhibited high overall accuracy but struggled to correctly classify smishing messages due to the skewed distribution of classes. These models improved their capacity to identify smishing instances more reliably by achieving a better balance among recall and precision using SMOTE. SMOTE was chosen over techniques like random under sampling and cost-sensitive learning because it generates synthetic examples rather than discarding data or requiring manual cost adjustments, thus preserving valuable information and enhancing generalizability. Additionally, under sampling risks losing important patterns in the majority class, while cost-sensitive learning demands fine-tuning of cost matrices, which can be complex and dataset-dependent. Therefore, SMOTE provided a practical and effective solution to improve model performance in this imbalanced classification scenario.

3.4 DATA SPLITTING

The train test split function of the Sklearn Python module is used to divide the dataset into a training set and a test set. There are 80% of the data points in the training set and 20% in the test set. After splitting, the sizes of both sets are checked to ensure the data has been correctly divided. The text is cleaned, tokenised, and balanced to ensure optimum performance since these stages prepare the dataset for future analysis and model training.

3.5 PROPOSED ARCHITECTURE OF DEEP LEARNING AND MACHINE LEARNING MODELS

XGBoost and Random Forest alongside CNN and RNN together with LSTM make up the list of models. The article introduces fundamental models with architectural details and operating principles along with essential features relevant to their practical applications. The models XGBoost, Random Forest, CNN, RNN, and LSTM were selected because it performs effectively for SMS phishing detection tasks. The ensemble learning methods XGBoost and Random Forest demonstrate superior capability to process imbalanced data with effectiveness. Local text relations which CNN identifies in textual data make it an effective tool for detection of phishing keywords. Sequence-based learning functions of RNN along with its similar model LSTM helps users understand contextual message relationships. Selection of these models focused on identifying models able to generalize well with improved classification accuracy levels. The following models are discussed below why we used with advantages:

- **Random Forest:** An ensemble learning algorithm, RF is basically a combination of multiple DT for better classification and outperform overfitting. It works very well with structured text features, and it performs well in discriminating between smishing and

real messages. The model was chosen for its outstanding ability to generalise to new data, rank features according to their significance, and handle noisy inputs with resilience.

- **XGBoost:** XGBoost or known as Extreme Gradient Boosting is boosting algorithm which enhances the decision trees through gradient boosting. It is perhaps one of the most popular modes of handling imbalanced datasets since it is fast, accurate and efficient. Therefore, the study employs XGBoost primarily because of its capability of handling non-linear features, sparsity, and high-classification models that are essential in a detection of phishing.
- **Convolutional Neural Network (CNN):** The CNNs serve efficiently in image processing tasks and operate effectively in text classification tasks. The convolutional filters establish word spatial relations while recognizing text patterns within local word groupings. The usage of CNNs in this study is since these models can employ contextual information analysis to differentiate between genuine and phishing communications.
- **Recurrent Neural Network (RNN):** One of RNN models' strengths is their ability to analyse sequential data, which helps them thrive in NLP applications. Due to their distinct design RNNs store earlier message inputs so it can establish relationships between message words. RNNs enable the study to use their sequential learning capabilities for understanding phishing message patterns through structure analysis.
- **Long Short-Term Memory (LSTM):** Text data benefits greatly from the special capabilities of LSTMs which derive from their origin as RNNs. These models solve the gradient disappearance issue found in customary RNNs which enhances their utility for text-based operation. LSTMs operate in the research to identify patterns in extended text which enhances recognition of complex attacks that extend over multiple words.

3.5.1 Hyperparameter tuning of CNN, RNN, and LSTM

The parameters are batch size=32, loss="binary cross entropy," optimizer="Adam," and epochs=10. Four different types of layers are commonly seen in CNNs: convolutional, pooling, activation function, and fully connected.

Batch Size=32: The batch size=32 parameter in machine learning defines how many samples are processed before updating the models weights. The dataset is fed in smaller portions of 32 samples each rather than the whole dataset all at once. This approach balances memory efficiency and training stability. Learning using a lower batch size could be more stable, but it might take more time. Training with a bigger batch size is faster, but it uses more memory.

Adam Optimizer: Adams approach, which stands for Adaptive Moment estimation, was utilised throughout a training phase to achieve optimisation with a total of 10 epochs (Kingma & Ba, 2015). Equations (3.1) -(3.4) specify the mathematical notations for Adam, where η is the pace of initial learning, g_t determines the gradient along ω_j at time t , x_t calculates the average gradient along ω_j exponentially, δ_1 is the mean squared value of the gradient along ω_j , expressed as an exponential, and δ_1, δ_2 are hyperparameters.

$$x_t = \delta_1 * x_{t-1} - (1 - \delta_1) * g_t \dots \dots \dots (3.1)$$

$$y_t = \delta_2 * y_{t-1} - (1 - \delta_2) * g_t^2 \dots \dots \dots (3.2)$$

$$\Delta\omega_t = -\eta \frac{x_t}{\sqrt{y_t + \epsilon}} * g_t \dots \dots \dots (3.3)$$

$$\omega_{t-1} = w_t + \Delta\omega_t \dots \dots \dots (3.4)$$

The starting learning rate should be set at 0.001 right now. To account for epoch-specific variations in loss values, we divide the learning rate by 10. As a precaution against overfitting, the dropout rate is kept at 0.5 for the whole model training process (Srivastava et al., 2014). After that, we make a note of the optimal model weights determined by the validation loss minimization.

Binary cross entropy: The binary cross entropy is a common loss function for binary classification applications such as phishing detection that aim to sort data into two groups: spam and ham (Minh & Nguyen, 2019). It finds a discrepancy among a models projected probabilities (\hat{y}) and an actual label (y). A function penalizes incorrect predictions, especially when the confidence is high but incorrect. For a single instance as Equations 3.5:

$$L = -\frac{1}{N} \sum_{i=1}^N (y \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)) \dots \dots \dots (3.5)$$

Where:

- y : - true label (0 or 1).
- \hat{y}_i : Predicted probability of the positive class ($0 \leq \hat{y}_i$).

Model: "sequential_11"

Layer (type)	Output Shape	Param #
conv1d_6 (Conv1D)	(None, 98, 32)	128
max_pooling1d_6 (MaxPooling1D)	(None, 49, 32)	0
conv1d_7 (Conv1D)	(None, 47, 64)	6,208
max_pooling1d_7 (MaxPooling1D)	(None, 23, 64)	0
flatten_3 (Flatten)	(None, 1472)	0
dense_22 (Dense)	(None, 128)	188,544
dense_23 (Dense)	(None, 2)	258

Total params: 195,138 (762.26 KB)
 Trainable params: 195,138 (762.26 KB)
 Non-trainable params: 0 (0.00 B)

Figure 8. Model summary of CNN

Figure 8 presents the model summary of a CNN named "sequential_11," designed for a binary classification task. The architecture consists of two convolutional layers (conv1d_6 and conv1d_7) with max-pooling layers (max_pooling1d_6 and max_pooling1d_7) for feature extraction and dimensionality reduction. As pattern complexity increases, the number of filters in every convolutional layer grows from 32 in the first to 64 in the second. The output is then flattened (flatten_3) and passed through two dense layers (dense_22 and dense_23). A binary categorisation is shown by the last dense layer's two output neurones, one for each class. "Trainable params" and "non-trainable params" numbers show that every one of the model's 195,138 parameters can be trained. The total size of the parameters is 762.26 KB.

Model: "sequential_12"

Layer (type)	Output Shape	Param #
simple_rnn_2 (SimpleRNN)	(None, 64)	4,224
dense_24 (Dense)	(None, 128)	8,320
dropout_5 (Dropout)	(None, 128)	0
dense_25 (Dense)	(None, 2)	258

Total params: 12,802 (50.01 KB)
 Trainable params: 12,802 (50.01 KB)
 Non-trainable params: 0 (0.00 B)

Figure 9. Model summary of RNN

RNN model "sequential_12," built for a binary classification job, is shown in Figure 9. The model begins with a Simple RNN layer (simple_rnn_2) with 64 units, which processes sequential input and produces a 64-dimensional output. The feature representation is then

increased by adding a dense layer (dense_24) with 128 neurones. To further avoid overfitting, a dropout layer (dropout_5) is used, which has an assumed non-zero dropout rate but is not explicitly stated. Finally, a dense output layer (dense_25) with 2 neurons is used for binary classification. The model requires 50.01 KB of storage space due to its 12,802 trainable parameters.

Model: "sequential_13"

Layer (type)	Output Shape	Param #
lstm_8 (LSTM)	(None, 64)	16,896
dense_26 (Dense)	(None, 128)	8,320
dense_27 (Dense)	(None, 2)	258

Total params: 25,474 (99.51 KB)
 Trainable params: 25,474 (99.51 KB)
 Non-trainable params: 0 (0.00 B)

Figure 10. Model summary of LSTM

Figure 10 presents the model summary for an LSTM-based neural network named "sequential_13," designed for binary classification. A 64-unit LSTM layer (lstm_8) and two dense layers (dense_26 and dense_27) make up the model. For binary classification, there are two layers of dense layers: one with 128 neurones and the other with 2. With a total of 25,474 trainable parameters, the model's size comes to 99.51 KB.

3.6 COMPUTATIONAL EFFICIENCY OF MODELS: MODEL EFFICIENCY AND REAL-WORLD FEASIBILITY

To evaluate the practical deployment potential of the models, we analyzed training time, memory usage, and inference speed. XGBoost and Random Forest showed faster training times and lower memory consumption compared to DL models, making them suitable for lightweight applications such as deployment on mobile networks. CNN, RNN, and LSTM, while effective in terms of accuracy, required more computational resources and longer inference times, indicating limitations for real-time use on resource-constrained devices. This comparison emphasizes the real-world feasibility of tree-based models for scalable and efficient smishing detection systems.

3.6.1 Model justification

In this study, models like XGBoost, Random Forest, CNN, RNN, and LSTM were selected due to their proven efficacy in SMS phishing detection tasks. The better-balanced trade-off among performance and computational economy offered by older transformer-based models like BERT

and Roberta was favoured by CNN, RNN, and LSTM. LSTM is well-suited for phishing detection since it circumvents the shortcomings of conventional RNNs when it comes to dealing with long-term dependencies, in contrast to CNN and RNN, which excel at capturing sequential and local correlations in text data, respectively. Although transformer-based models like as BERT are quite good at text categorisation, we didnt include them since training and fine -tuning them requires a lot of computing power, which could be too much in low-resource settings. Moreover, BERT and Roberta’s larger model size could lead to slower inference times, limiting their real-time application in phishing detection. Future research could explore the integration of transformer-based models, focusing on their adaptation to smaller-scale environments or investigating their performance in hybrid architectures, where the strengths of both traditional and transformer-based models are leveraged for enhanced phishing detection accuracy.

3.7 MODEL EVALUATION METRICS FOR SMS SMISHING DETECTION

A performance of the ML model is assessed after training it to identify SMS smishing using important classification metrics like accuracy, precision, recall, F1score, and the ROC-AUC score. Additionally, a confusion matrix is utilized to offer a thorough breakdown of predictions.

Chapter 4 Experimental Results

This chapter offers experimental dataset analysis, assessment measures, and classification performance. The accuracy, precision, recall, and F1score are highlighted in a comparison of ML and deep learning models. Limitations, misclassification impact, and potential improvements are also discussed.

4.1 DATASET ANALYSIS RESULTS

Data visualisation tools serve data analysis by converting complicated datasets into visual descriptions. The analysis of data requires multiple reasons to depend on data visualization tools. The analysis process benefits from such capabilities because analysts can observe data patterns that prove difficult to identify without these tools. Graphs, charts, dashboards, and maps are examples of visual representations that can aid analysts in identifying anomalies or outliers and understanding the underlying data. Furthermore, data visualisation technologies improve the conveyance of complex information to a larger audience, whichever their degree of knowledge(Lavanya et al., 2023).

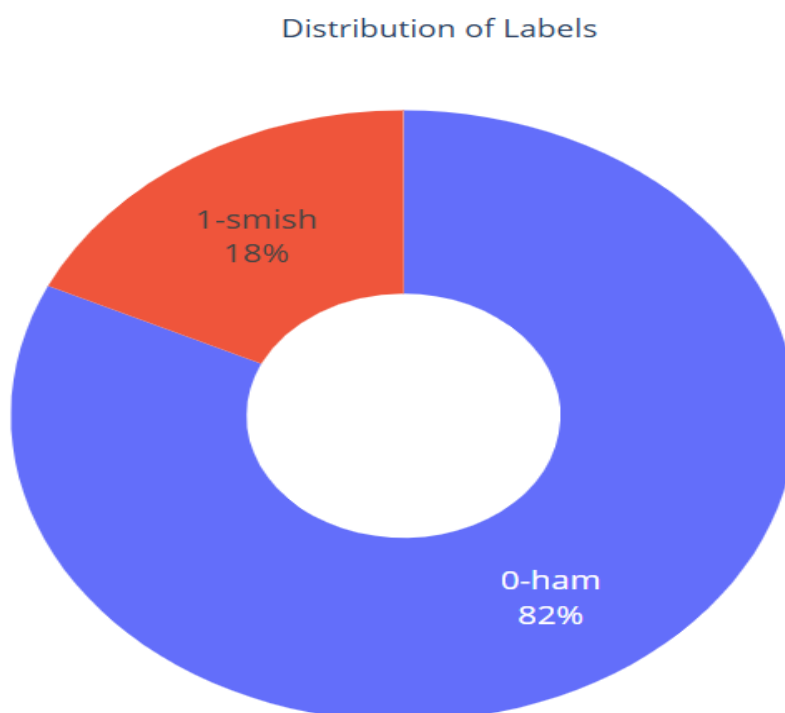


Figure 11. Distribution of Labels

Figure 11 shows a donut chart "Distribution of Labels" with two segments before balancing. The larger segment, shown in blue, represents "0-ham" and makes up 82% of the

"claim," "congratulation," "offer," and "urgent", as these represent common words used by criminals in phony SMS messages to trick potential victims.

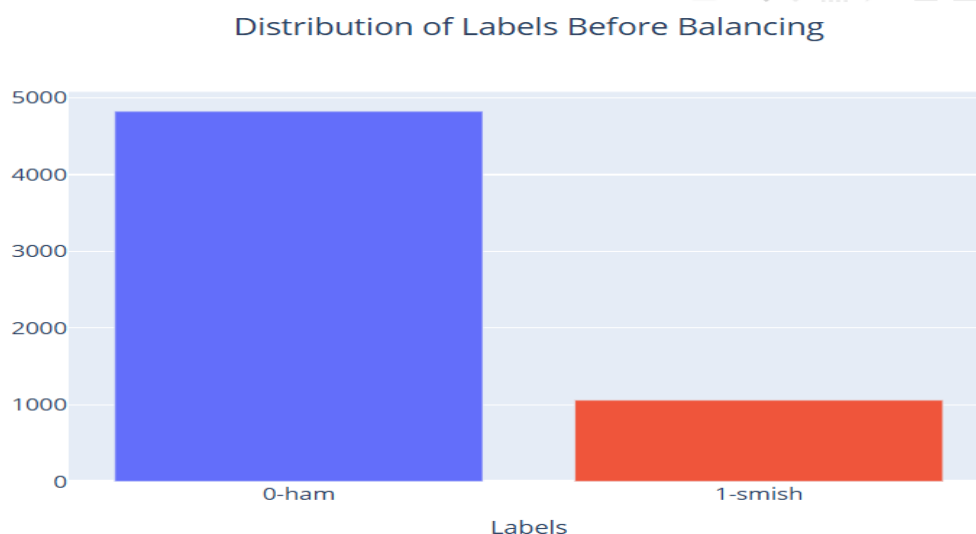


Figure 14. Distribution of labels before balancing

Figure 14 displays a bar chart of a class distribution before applying any balancing techniques. A major imbalance among the "ham" and "smish" classes may be detected using this method.

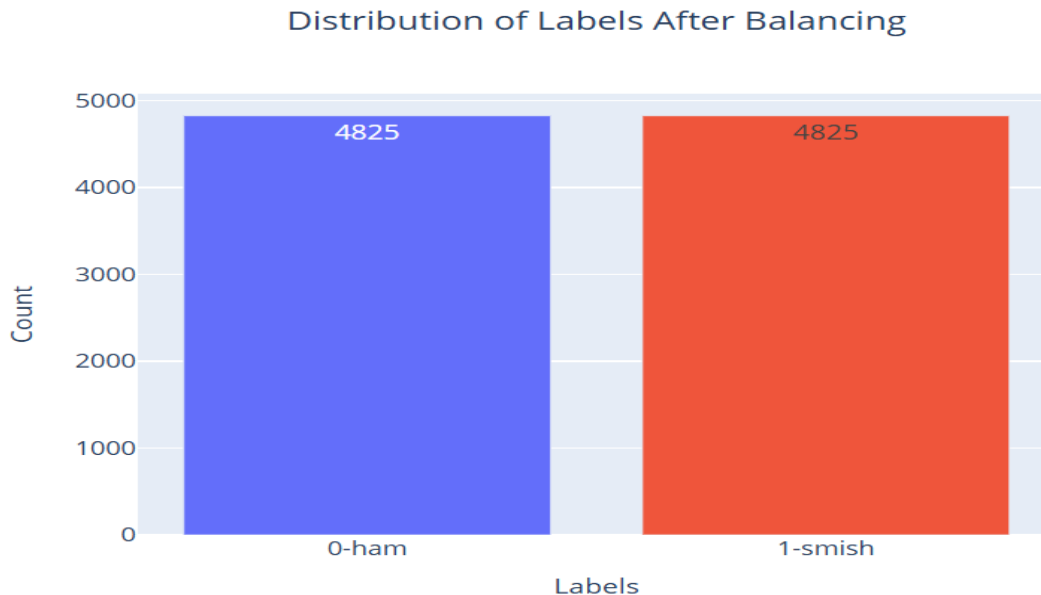


Figure 15. Distribution of labels after balancing

Figure 15 displays a distribution of labels after balancing (4825 for both classes). The SMOTE is applied to address a class imbalance. To make a dataset more representative of the minority class, SMOTE creates synthetic examples of "smish" texts.

4.2 CLASSIFICATION RESULTS

4.2.1 XGBoost classifier

Accuracy	Precision	Recall	F1 score
97.05%	96.04%	97.98%	97%

Table 2. XGBoost classifier Model

Table 2 displayed the outcomes of the XGBoost classifier model. A result how that XGBoost classifier achieves 97.05% accuracy, 96.04% precision, 97.98% re-call, and 97 % F1 score.

```

Classification Report for Test Set:
              precision    recall  f1-score   support

   ham         0.98         0.96         0.97         990
   smish        0.96         0.98         0.97         940

 accuracy              0.97         0.97         0.97         1930
 macro avg              0.97         0.97         0.97         1930
 weighted avg           0.97         0.97         0.97         1930

```

Figure 16. Classification report of XGBoost classifier Model

Figure 16 displayed an outcome of the XGBoost Classifier model's evaluation on a test set of 1930 cases for classification. The model's total accuracy of 97% is a testament to its excellent accuracy. Specifically, it performs with high precision, recall, and F1-score 0.97 for both classes, 'ham' (990 instances) and 'smish' (940 instances), indicating a balanced and robust classification capability. The model's constant and good performance across both classes without any substantial bias is shown by the weighted and macro averages, which demonstrate 0.97 across all measures. These findings are further corroborated by the data.

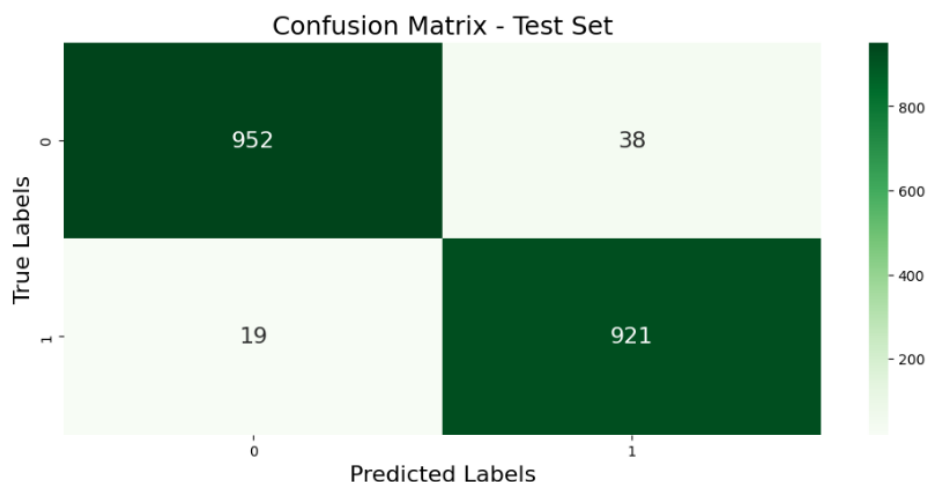


Figure 17. Confusion Matrix of XGBoost classifier

Figure 17 displays the Confusion Matrix of the XGBoost Classifier. Its TP count is 952, its false positive count is 38, its false negative count is 19, and its true negative count is 921.

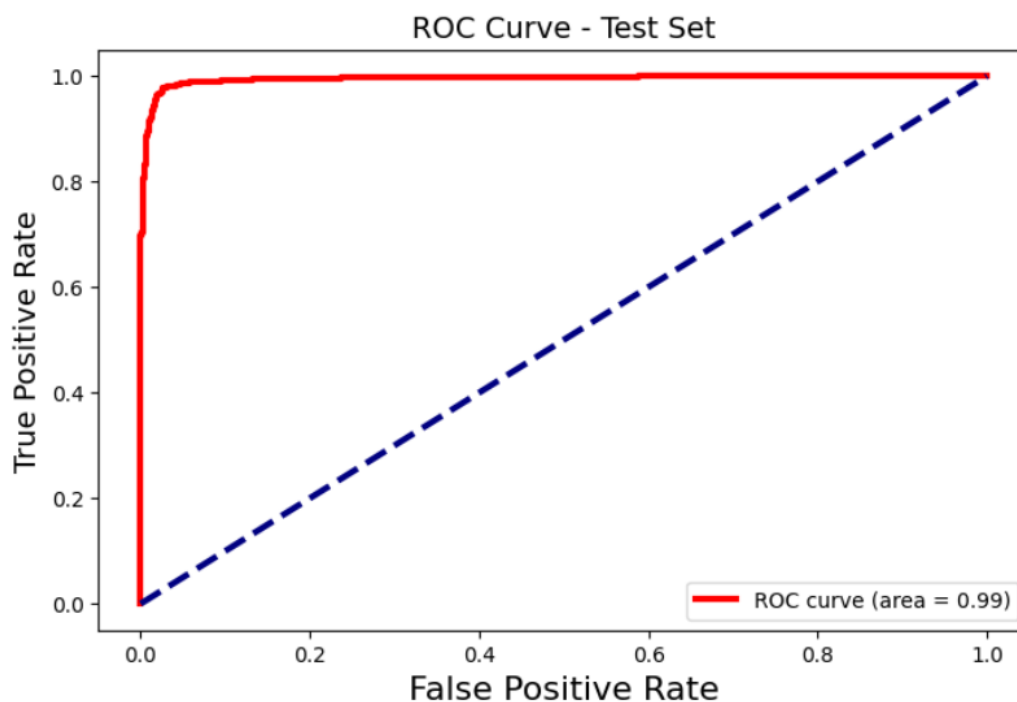


Figure 18. ROC Curve of XGBoost model

Figure 18 displayed the ROC curve for a test set which depicts a model's classification ability. The ROC score of 0.99 appears on the red solid line which connects points in Figure along with a blue dashed line serving as the random classifier baseline (diagonal line from 0,0 to 1,1). An AUC of 0.99 indicates almost perfect discrimination capacity beyond random categorisation, and the classifier shows exceptional performance in distinguishing among positive and negative case classifications.

4.2.2 Random Forest classifier

Accuracy	Precision	Recall	F1 score
96.63%	96.89%	96.17%	96.53 %

Table 3. Random Forest classifier Model

Table 3 displayed an efficiency of RFC, with 96.63% accuracy, 96.89% precision, 96.17% recall, and 96.53% f1 score.

Classification Report for Test Set:

	precision	recall	f1-score	support
ham	0.96	0.97	0.97	990
smish	0.97	0.96	0.96	940
accuracy			0.97	1930
macro avg	0.97	0.97	0.97	1930
weighted avg	0.97	0.97	0.97	1930

Figure 19. Classification report of Random Forest classifier

Figure 19 depicts RFC classification results. For the gammon category, the model achieved an F1score of 0.96, a precision of 0.97, and a re-call of 0.97. When it came to the Smish class, the re-call was 0.96, the precision was 0.97, and the F1-score 0.96. Overall, a model does quite well, with an accuracy of 0.97.

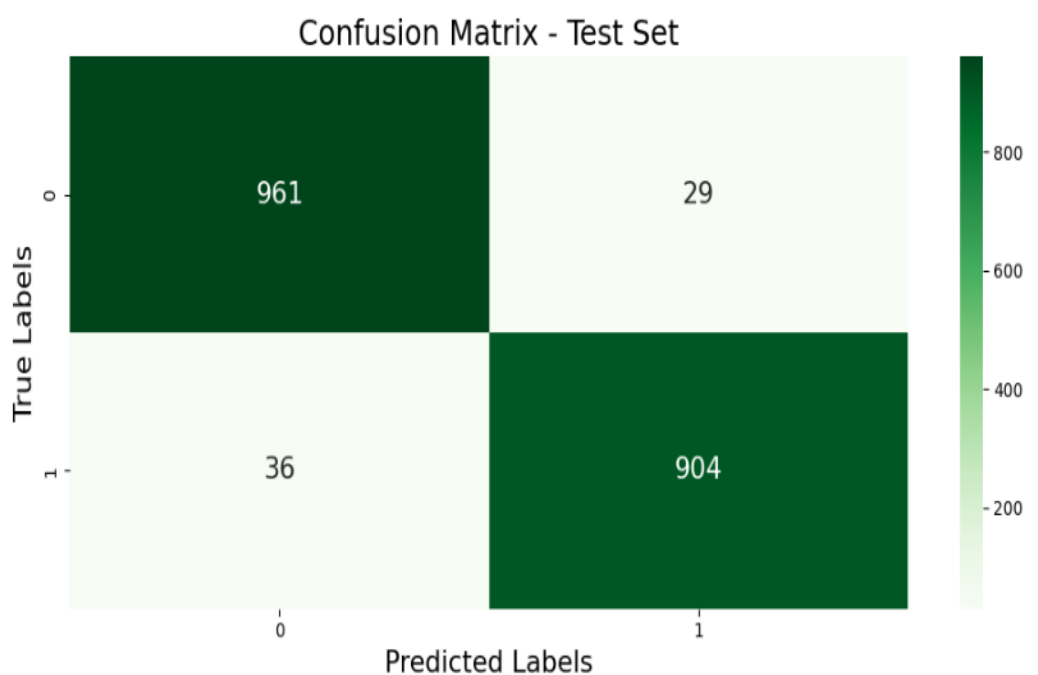


Figure 20. Confusion matrix of Random Forest classifier

Figure 20 displayed a RFC's Confusion Matrix, which contains 961 TP, 29 FP, 37 FN, and 904 TN.

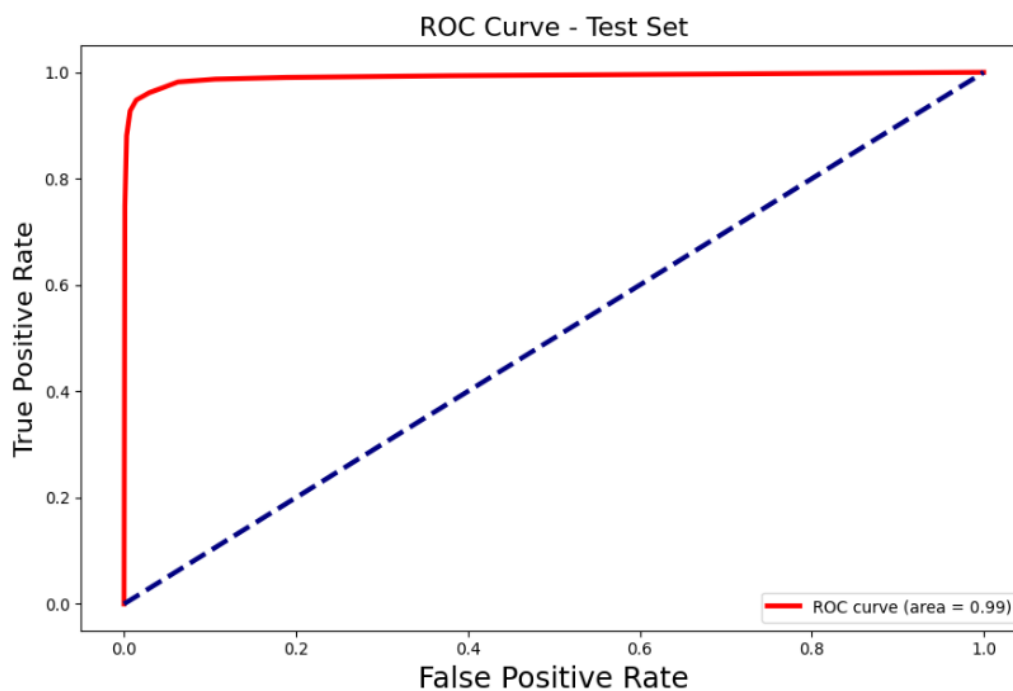


Figure 21. Roc Curve of Random Forest Classifier

Figure 21 shows RFC ROC curve. The ROC curve from a model reaches excellent AUC performance of 0.99 while showing a rapid increase at zero false positives and reaching plateau at very high true positives. The model exhibits significant performance superiority over random classification by demonstrating its results through the blue dashed diagonal line.

4.2.3 CNN Model

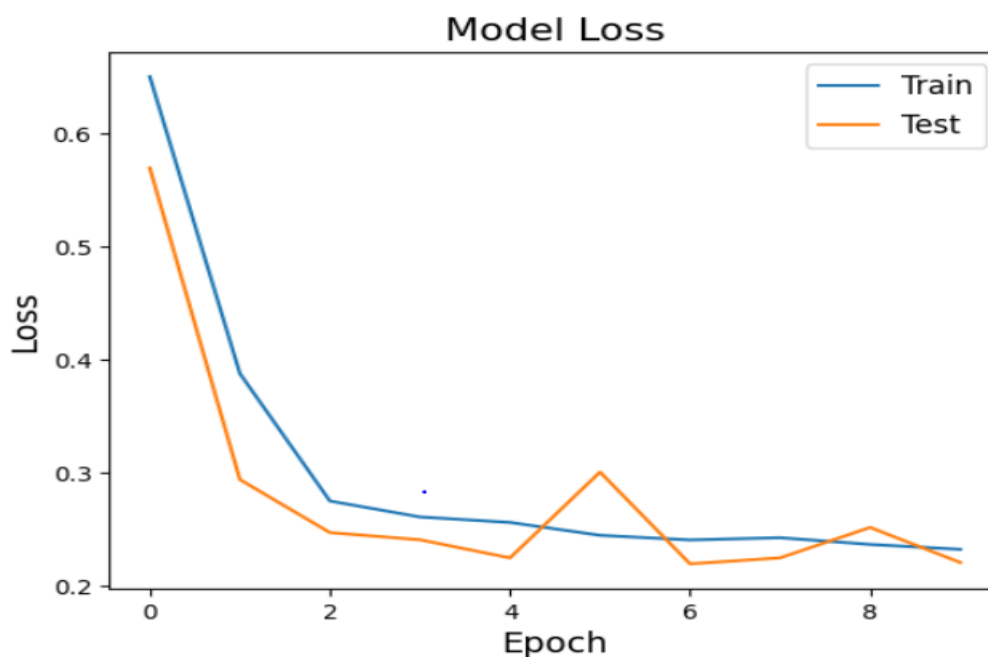


Figure 22. Model loss graph of CNN model

Figure 22 demonstrates how training and testing loss proceeded throughout 8 epochs for the model. The training loss line along with the testing loss line begin at values of approximately 0.52 and 0.37 during their initial phases before converging to similar values of 0.22 at epoch 6.

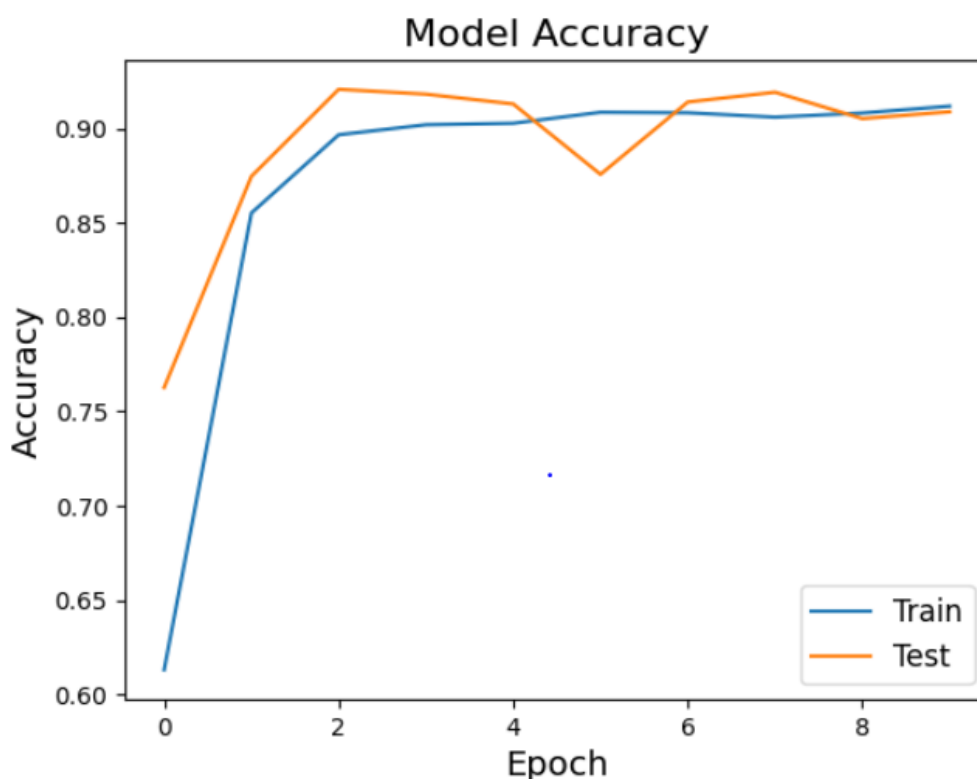


Figure 23. Model accuracy graph of train and test model

The accuracy plot of Figure 23 demonstrates the training accuracy alongside testing accuracy for 8 epochs. The starting value for training accuracy sits at 0.70 before it shows quick improvement during the initial two epochs yet test accuracy begins at 0.91. Between epochs 6 and 8 both curves show no measurable deviation from 0.92 accuracy before reaching equilibrium. Model learning demonstrates excellent performance because testing accuracy of 0.9088 maintains stability while training accuracy of 0.9113 approaches the initial values.

Accuracy	Precision	Recall	F1 score
90.88	94.01	86.81	90.27

Table 4. Performance of CNN model

Table 4 show a CNN model with 90.88accuracy, 94.01% precision, 86.81recall, and 90.27% f1 score.

Classification Report for Test Set (CNN):				
	precision	recall	f1-score	support
0	0.88	0.95	0.91	990
1	0.94	0.87	0.90	940
accuracy			0.91	1930
macro avg	0.91	0.91	0.91	1930
weighted avg	0.91	0.91	0.91	1930

Figure 24. Classification report of CNN model

The CNN model's categorisation report is displayed in Figure 24. Compared to Smish, Ham had 0.88 precision, 0.95 recall, and 0.91 F1-score, while Ham had 0.94 precision, 0.87 recall, and 0.90 F1-score. The model has an average accuracy rate of 0.91.

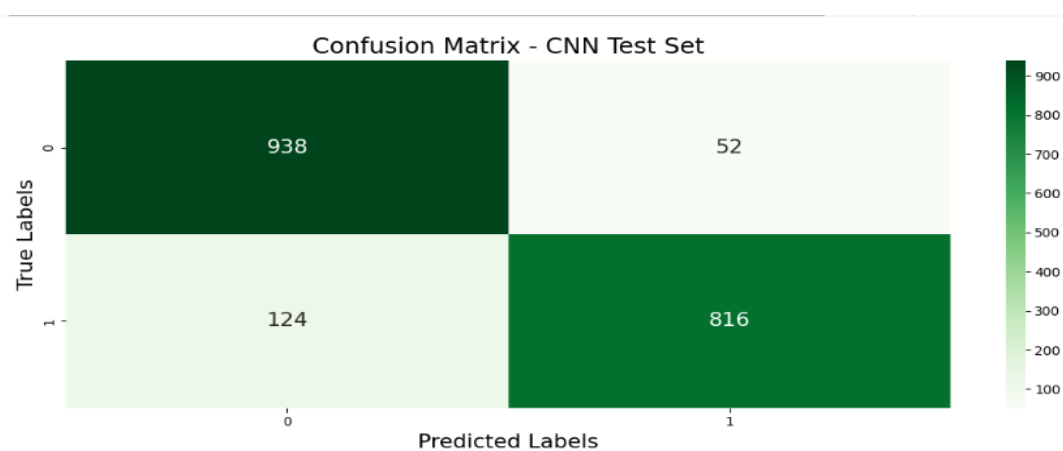


Figure 25. Confusion matrix of CNN model

Figure 25 displayed the CNN confusion matrix with 938 TP, 124 FP, 52 FN, and 816 TN.

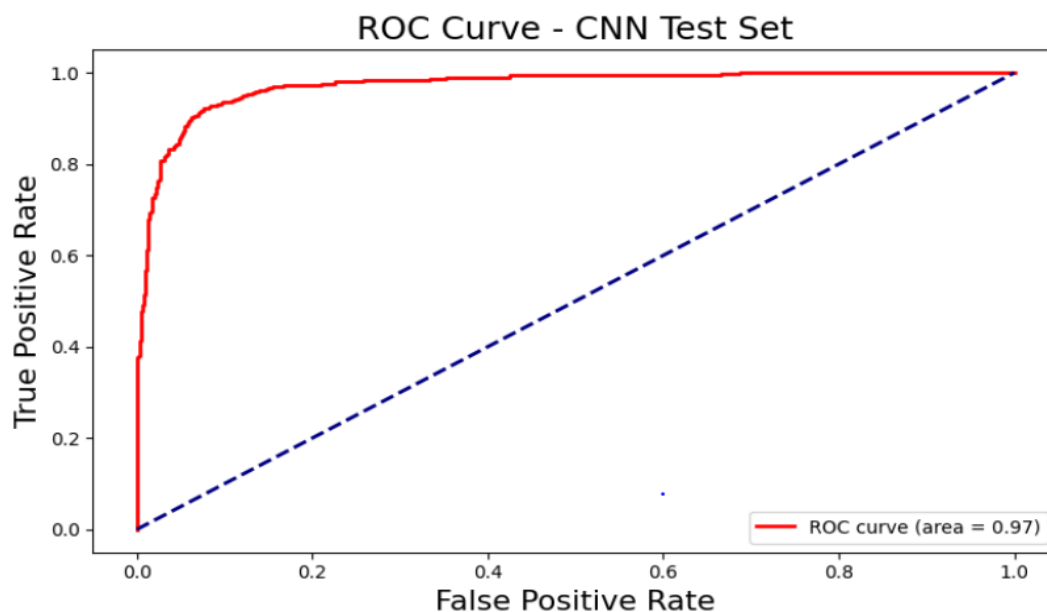


Figure 26. ROC Curve of CNN model

Figure 26 shows the ROC curve, which is used to quantify the performance of a CNN model on a test set. Because it increased sharply at first and then quickly stabilised at a high TPR, the performance's red curve shows an AUC value of 0.98. At baseline the random classifier performance aligns with the blue dashed diagonal line.

4.2.4 RNN Model

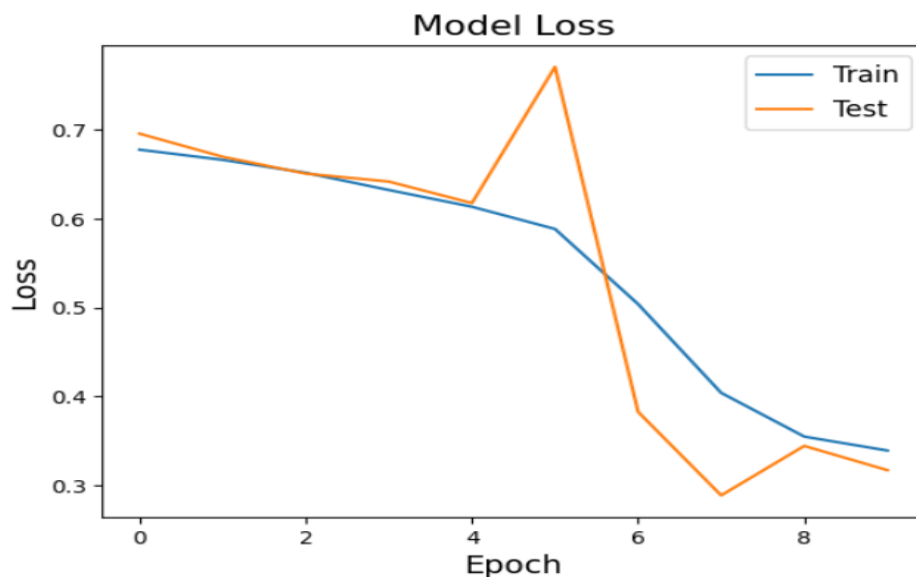


Figure 27. RNN Model Loss graph of Train and test model

Figure 27 displays the training loss (blue) and test loss (orange) during 8 epochs in which both series started from 0.67 and showed steady reduction until the test loss achieved its minimum point of 0.40.

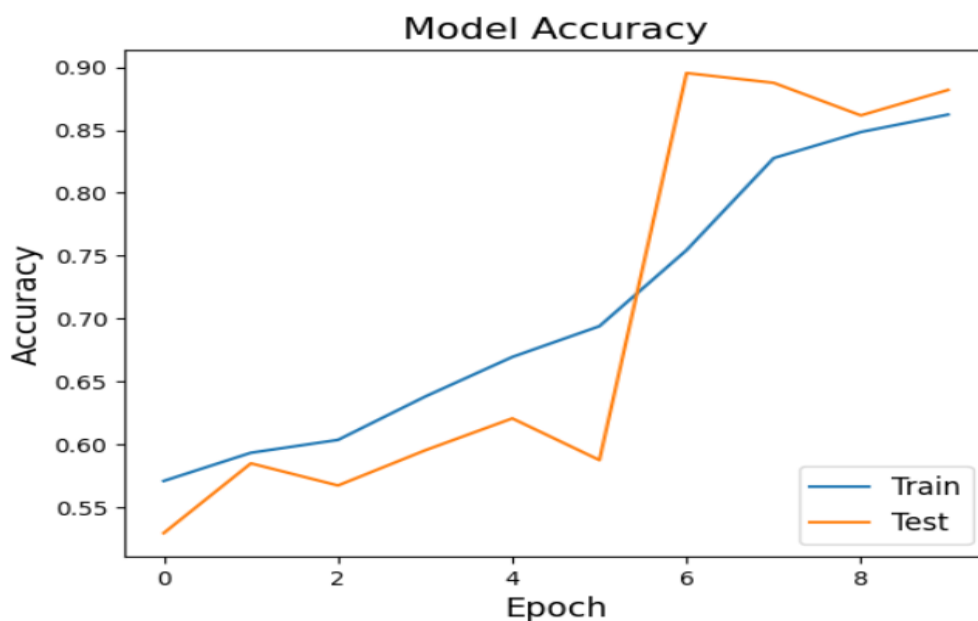


Figure 28. RNN Model Accuracy graph of Train and test model

Figure 28 shows a RNN model accuracy graph through 9 epochs shows training accuracy (blue) and test accuracy (orange) beginning at approximately 0.57. The accuracy brings mixed results during the 9 epochs as test accuracy reaches its highest point at 0.85 during epoch 8 before declining slightly but training accuracy continues to grow up to its peak at around 0.77 at epoch 8.

Accuracy	Precision	Recall	F1 score
88.19%	83.71%	94.04%	88.58%

Table 5. RNN model performance

Table 5 demonstrate an RNN model with 8.19% accuracy, 83.71% precision, 94.04% recall, and 88.58% F1 score.

```

Classification Report for Test Set (RNN):
              precision    recall  f1-score   support

     0         0.94         0.83         0.88         990
     1         0.84         0.94         0.89         940

 accuracy              0.88         1930
 macro avg              0.89         0.88         0.88         1930
 weighted avg           0.89         0.88         0.88         1930

```

Figure 29. Classification report of RNN model

Figure 29 displays the RNN classification results. For identifying "ham" text, the model showed a f1score of 0.94, precision of 0.83, and recall of 0.88. The "smish" class achieved a f1score of 0.88 with an accuracy precision of 0.94 and a recall of 0.88. The model has a total accuracy of 0.88.

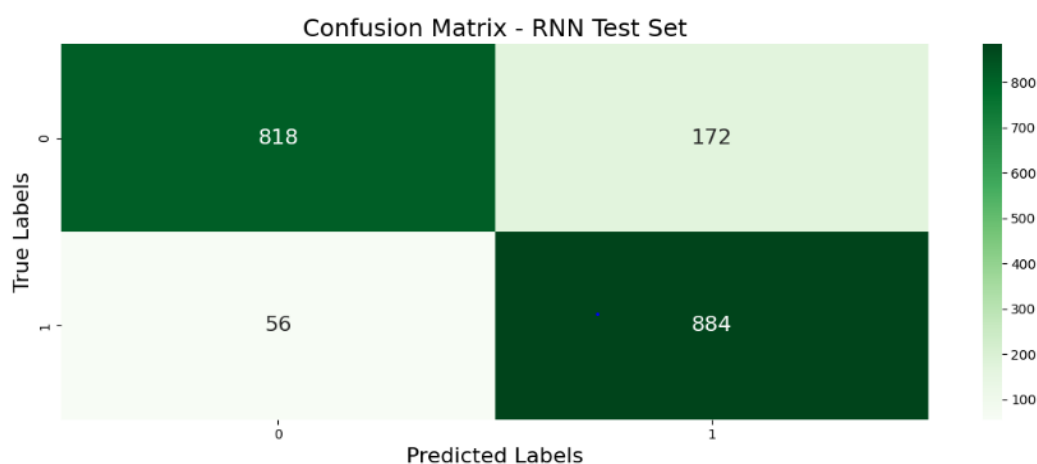


Figure 30. Confusion matrix of RNN model

Figure 30 displayed the RNN Confusion Matrix, which displays 818 TP, 56 FP, 172 FN, and 884 TN.

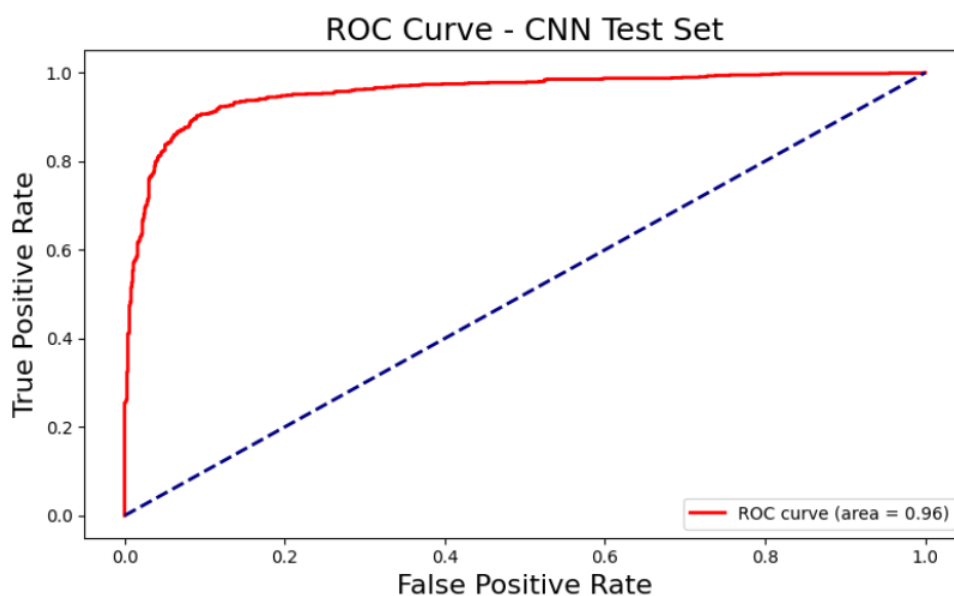


Figure 31 ROC curve of the RNN model

Figure 31 displays a RNN model's ROC curve. A model's AUC score of 0.96 indicates that it performs exceptionally well in classification, as evidenced by the red curve's sharp ascent from the origin while remaining above the blue diagonal baseline that represents random classification. A steep ROC curve indicating great discrimination with low FPR and high TPR suggests that the model can accurately distinguish between classes.

4.2.5 LSTM Model

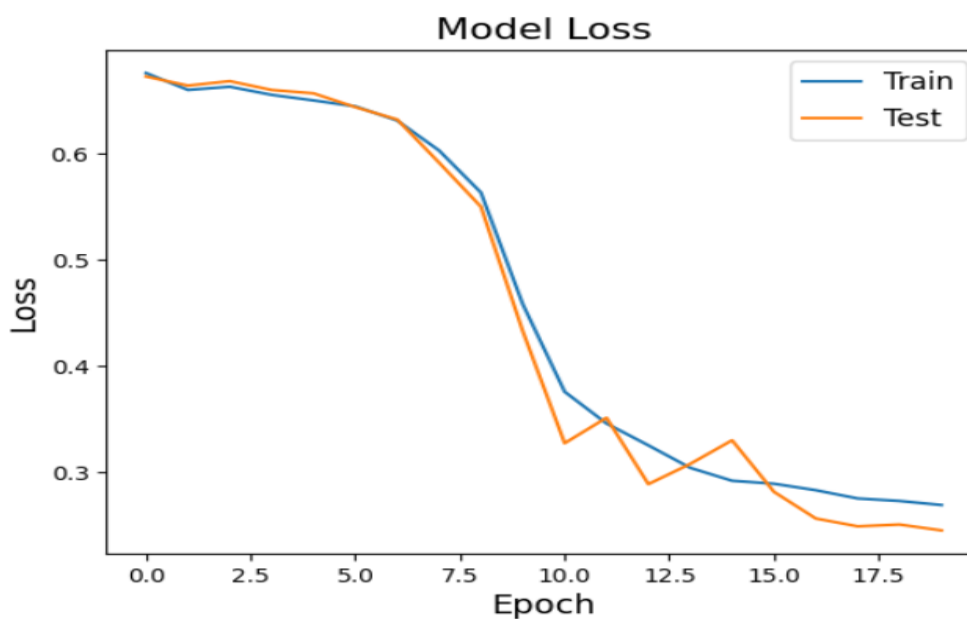


Figure 32. Loss graph of LSTM model

The figure 32 presents model loss analysis between training (blue) and testing (orange) during epoch progression. The losses maintain a steady state until it begins to decrease at epoch 8. During training the model loss fluctuation occurs before establishing stability while the training loss reduces steadily this suggests both learning advancement and potential overfitting issues or instabilities.

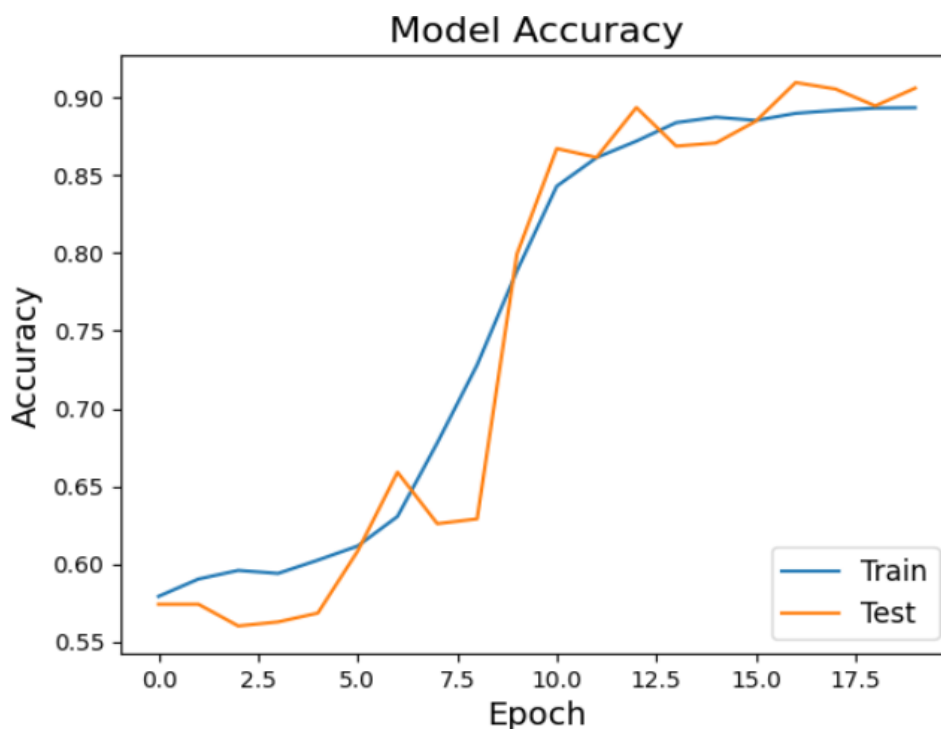


Figure 33. Accuracy graph of LSTM model

Figure 33 shows the accuracy performance between training blue lines and testing orange lines across different epoch numbers. During the first stages both accuracy rates increase smoothly until the 9th epoch represents a major boost followed by unsettled fluctuations. The training accuracy continued rising steadily before achieving its highest value around the last epochs. The test precision demonstrates an upward pattern containing occasional fluctuations which demonstrates solid generalization capabilities even though there are minor variances.

Accuracy	Precision	Recall	F1 score
90.62%	92.98%	87.34%	90.07%

Table 6. LSTM Model Performance Results

Table 6 demonstrate that LSTM model has 90.62% accuracy, 92.98% precision, 87.34% recall, and 90.07% F1 score.

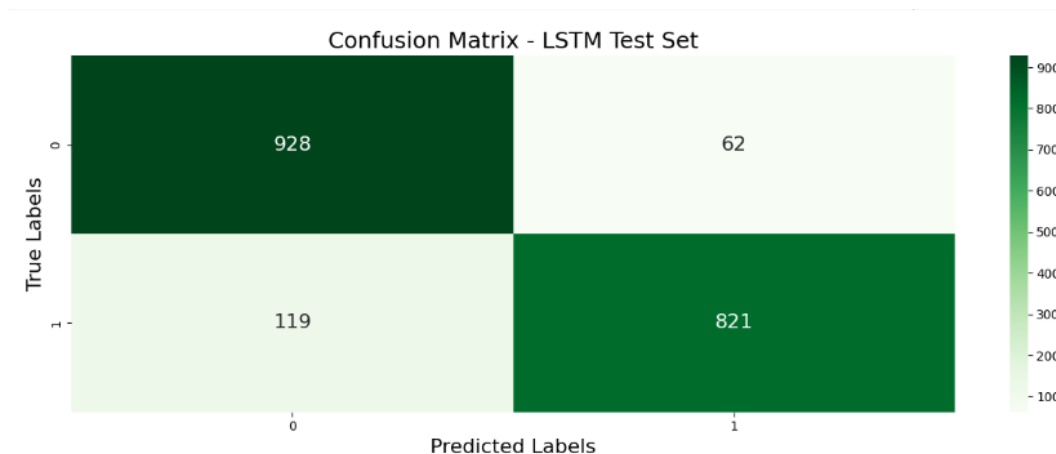


Figure 34. Confusion matrix of LSTM model

Figure 34 displayed LSTM model confusion matrix with TP of 928, FP of 119, FN of 62, and TN of 821.

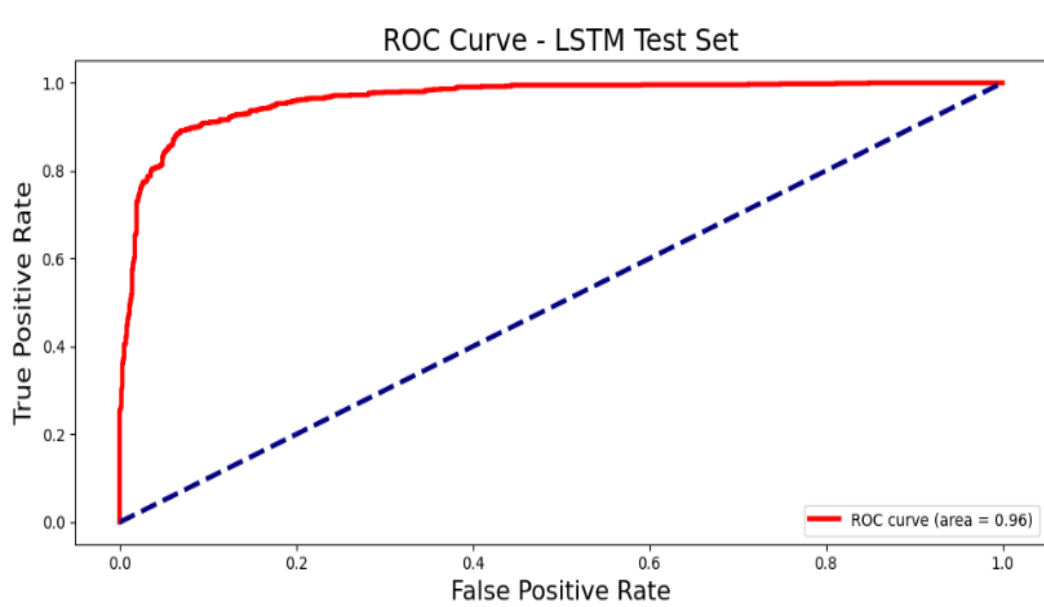


Figure 35. The LSTM model's ROC curve

The LSTM model's ROC curve for the test set performance is displayed in Figure 35. A model performs exceptionally well in classification tasks since its red curve starts sharply and raises significantly above the blue baseline. The AUC score reaches 0.96. The model displays strong discrimination abilities and optimal discrimination precision according to the assessment results.

4.3 COMPARATIVE ANALYSIS

The accuracy, precision, recall, and F1 score of the ML and the DL models examined previously are compared in this section.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
XGBoost	97.05	96.04	97.98	97.00
Random Forest	96.63	96.89	96.17	96.53
CNN	90.88	94.01	86.81	90.27
RNN	83.71	94.04	88.58	83.71
LSTM	92.98	87.34	90.07	92.98

Table 7. Comparison of the ML and DL models under study

Table 7 compares several ML and DL models according to F1 score, recall, accuracy, and precision. Among a model, XGBoost demonstrates a highest performance with an accuracy of 97.05%, precision of 96.04%, recall of 97.98%, and an F1 score of 97.00%, making it the most effective model overall. RF follows closely with an accuracy of 96.63%, achieving slightly higher precision (96.89%) but lower recall (96.17%) compared to XGBoost, finding in an F1-score of 96.53%. Among the DL models, LSTM outperforms CNN and RNN with an accuracy of 92.98%, a precision of 87.34%, a recall of 90.07%, and an F1score of 92.98%, indicating strong sequential learning capability. CNN, with an accuracy of 90.88%, excels in precision (94.01%) but has a relatively lower recall (86.81%), leading to an F1-score of 90.27%. Meanwhile, RNN exhibits the lowest accuracy (83.71%), despite having a high precision of 94.04% and recall of 88.58%, showing that while it predicts positive cases well, it struggles with overall classification balance, as reflected in its F1-score of 83.71%. The outcomes display that LSTM is the greatest DL method, and that XGBoost and RF are the most trustworthy models.

4.3.1 Limitations of the Current Approach and Improvement

The presented methods encounter significant drawbacks due to imbalanced data distribution. Originally, the dataset contained few "smish" messages than "ham" messages which may result in biased model predictions. The SMOTE is employed as a solution to this problem. SMOTE introduces artificial noise in data when used so it reduces a model's capacity to make valid predictions on actual world scenarios. An intended model robustness can be improved through alternative resampling approaches that include Adaptive Synthetic Sampling (ADASYN) and cost-sensitive learning techniques.

The dependence on XGBoost and Random Forest traditional machine learning models presents a limitation since it fails to efficiently understand the contextual content of SMS messages. Among the tested DL models, RNN, CNN and LSTM provided diverse results where XGBoost achieved superior accuracy compared to these other models. An NLP deep learning model that combines transformer-based models with BERT would be ideal for such jobs. The Transformer design incorporates the capacity to comprehend the bidirectional context of words in a phrase by taking into account both the left and right contexts of a word at the same time. On the other hand, hybrid methods that combine DL with ML might provide superior categorisation outcomes.

When evaluating the model's efficacy on classification tasks, the key metrics used were F1 score, re-call, accuracy, and precision. The metrics serve to provide valuable information, but real-world implementations need security against adversarial examples together with resilience to changing phishing methods. Future advancements should incorporate adversarial training together with continual learning techniques which should run real-time detection systems that respond to new threats.

4.3.2 Impact of Misclassifications

SMS phishing detection misclassifications produce major consequences when dealing with security for sensitive applications. False classification incidents that identify legitimate messages as scams result in two negative impacts: user dissatisfaction and lost essential communication content. Business operations using automated systems face significant challenges when legitimate messages are misidentified as spam since this results in prevented communication with customers.

The danger is increased by false negatives, in which smishing communications are mistakenly categorised as authentic. Such errors can expose users to phishing attacks, financial fraud, and identity theft. Attackers continuously evolve their tactics, making it crucial for detection models to maintain high recall rates without compromising precision. Implementing more sophisticated anomaly detection mechanisms and periodic model updates can help mitigate these risks.

4.4 DISCUSSION

Traditional ML algorithms, XGBoost and RF, achieve superior results than deep learning methods CNN, RNN and LSTM in this text classification task. The ability of feature-based learning proved successful for SMS scam detection because short text message lengths restrict deep learning techniques that leverage contextual understanding. The restricted dataset size

restricts the complete achievement of deep learning models despite their typical requirement for extensive datasets. Researchers should use standardized future investigations by incorporating pre-trained language models together with data augmentation methods to enhance model generalization. The analysis generates important findings for SMS spam detection but needs more attention to become practical in actual deployments. The system's efficacy may be improved by integrating mobile security applications, adding user feedback mechanisms, and detecting events in real-time. Future research needs to study combined AI modelling approaches along with AI techniques and adversarial training methods to develop improved defence against evolving phishing attacks.

4.5 CRITICAL EVALUATION OF RESULTS

The results of the smishing detection models show high overall performance; however, a deeper critical evaluation reveals several areas for improvement. Misclassifications, particularly false negatives, were commonly observed when smishing messages mimicked legitimate SMS content using personalized language, shortened URLs, or grammatically correct text, making them harder to detect. Conversely, false positives often occurred with legitimate messages containing urgent tones or embedded links, which are also common in smishing. These misclassifications highlight the need for more context-aware models. Additionally, while DL models like CNN and LSTM outperformed traditional models in accuracy and precision, their performance varied with data imbalance and message complexity. Unexpectedly, some simple models occasionally performed comparably in recall, suggesting that complex architectures do not always guarantee superior detection. From a practical perspective, this research holds strong relevance for New Zealand, where the rise in mobile banking and e-commerce has made users increasingly vulnerable to smishing attacks. Therefore, deploying robust, adaptive detection systems can significantly enhance public digital safety and support national cybersecurity strategies.

Chapter 5 Conclusion and Future Work

Chapter 5 concludes the research in full, summarising its findings, outlining future work, and providing an overview of the creation and assessment of a New Zealand SMS phishing detection system.

5.1 CONCLUSION

The high demand of mobile technology combined with SMS messaging has resulted in a growing number of SMS-based phishing assaults known as "smishing" that endanger personal privacy and financial assets. These types of attacks have evolved in New Zealand and other areas beyond recognition with advanced sophistication making the detection of them more difficult. This study has tackled this critical cybersecurity problem by creating and testing automated detection systems that can identify and categorise smishing attempts using DL and ML.

The main research objective focused on developing an effective SMS phishing detection system designed for New Zealand conditions through the combination of global datasets with local precedent cases. This research concentrates on the development and assessment of ML and DL approaches for detecting smishing messages through SMS platforms resulting in valuable knowledge about automated threat analysis systems for the New Zealand environment. Multiple classification models were developed and analysed through this research to address rising SMS-based cyber threats by combining international examples from the Kaggle SMS Smishing Collection with real cases from New Zealand Department of Internal Affairs (DIA) anti-scam archive.

The initial phase established the dataset composed of 5,884 SMS messages between legitimate "ham" and fraudulent "smish" messages. Data pre-processing served as a vital foundation which integrated complex procedures to establish reliable and high-quality information for the models' success. The predictive models received data pre-processing that involved complete text normalization and advanced natural language processing methods including lemmatization. The SMOTE addressed the common cybersecurity dataset class imbalance problem to produce a balanced training dataset for the models.

ML and DL models demonstrated distinct patterns during their evaluation process for detection of smishing attacks. The XGBoost classifier demonstrated the highest performance

by reaching 97.05% accuracy and maintaining 96.04% precision coupled with 97.98% recall scores on the proposed dataset. XGBoost can handle difficult feature combinations while maintaining superior data imbalance management that leads to excellent detection results. The Random Forest classifier showed equally strong abilities by achieving 96.63% accuracy and similar precision and recall metrics while demonstrating ensemble methods' effectiveness.

The CNN model attained an accuracy of 90.88% and remarkable re-call and precision values, according to the deep learning findings. The CNN demonstrated excellence in handling information about text data spatial layout which generated better detection of refined phishing patterns. The implementation of RNN with LSTM models analysed temporal message aspects but produced slightly inferior outcomes compared to traditional ML with accuracy rates of 83.71% and 92.98% respectively.

The practical implementation analysis of these models produced significant findings which researchers presented through comparative studies. The traditional ML approaches XGBoost and Random Forest achieved superior performance by demonstrating better accuracy and computational resources compared to DL models. The limited number of entities (less than 6000 entities) and the brief message sizes in the dataset are the causes of this. The real deployment of SMS phishing detection systems requires traditional models due to their practical implementation suitability especially for resource-limited environments.

The study demonstrated the fundamental role that feature engineering with proper data representation plays in achieving successful model performance. Word2Vec embeddings showed effectiveness in term of semantic relationship comprehension thus enabling better message contextual and intentional comprehension. The machine learning models showed superior performance because it leveraged rich text data from these representations.

The outcomes of this research create valuable implications for cybersecurity standards particularly for mobile communication protection. According to research, ML detection systems are useful for spotting smishing attempts, thus cybersecurity experts and telecom companies may benefit from them. The successful approach to handle unbalanced datasets yields lesson learned for dealing with analogous difficulties in different cybersecurity implementation contexts.

The research extends knowledge about automated threat detection while delivering essential guidelines to establish SMS phishing detection systems in practice. Future development of cybersecurity-specific research will benefit from the systematic review of

modeling approaches together with their performance evaluations offered in this study. The evolutionary nature of SMS threats gives researchers a solid basis through this work to create state-of-the-art detection systems.

5.2 LIMITATIONS

Notwithstanding the smishing detection models' encouraging performance, a number of drawbacks affect their practicality. The use of pre-labeled datasets, which could not accurately reflect the changing linguistic patterns and strategies used in actual smishing assaults, is one significant drawback. Furthermore, while DL models like as CNN and LSTM have shown remarkable accuracy and precision, their implementation on devices with limited resources, such mobile phones, may be hampered by their high training data requirements and processing demands. The models also struggled with ambiguous messages and those written in mixed languages or local dialects, which are increasingly common in New Zealand's multicultural society. Furthermore, the black-box nature of DL limits transparency and interpretability, posing challenges for gaining user trust and regulatory approval. In real-world applications, these models would need to be continuously updated with new data, integrated with user feedback, and tested in dynamic environments to ensure robustness and adaptability to emerging threats.

5.3 FUTURE WORK

Future research should address the limitations mentioned above by integrating the developed SMS phishing detection models into real-world systems, especially in collaboration with telecom providers and banking institutions. This integration will allow for the assessment of the practical applicability and scalability of the models in operational settings. Furthermore, research into the incorporation of transformer-based models like BERT or Roberta, which have shown strong performance in various NLP tasks, should be pursued. While these models offer improved performance, their high computational requirements limit their real-time deployment in mobile networks. Future work should focus on optimizing or using lightweight versions of these transformer models to balance accuracy and efficiency. Finding solutions to the problem of model drift is another important area for prospective studies. Sustaining the models' efficacy over time requires ongoing retraining to account for new phishing techniques and developing patterns. Furthermore, improving the detection systems' resilience requires tackling the issue of adversarial assaults. Future research should also look at the privacy and ethical issues surrounding the use of these models, making sure that data protection laws are followed, and improving the security of sensitive data handled by these systems.

Future work will focus on improving the practical deployment of the proposed smishing detection system by addressing several real-world considerations. One key aspect is the integration of the model into telecommunications provider systems, which requires designing a lightweight and scalable architecture capable of real-time filtering without compromising system performance or user experience. This involves ensuring low latency and minimal memory usage to support deployment on mobile networks and edge devices. Moreover, as phishing tactics continuously evolve, it is essential to consider model drift and implement periodic retraining using up-to-date datasets to maintain detection accuracy over time. To further enhance robustness, future research should explore techniques to defend against adversarial attacks that attempt to manipulate SMS content to bypass detection. Privacy and data protection will also be crucial, particularly in handling sensitive user data during model training and prediction. Overall, a sustainable deployment strategy must include regular maintenance, automated updates, and secure integration pipelines to ensure the model remains effective in dynamic and potentially hostile environments.

References

- Abayomi-Alli, O., Misra, S., & Abayomi-Alli, A. (2022). A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset. *Concurrency and Computation: Practice and Experience*, 34(17), 1–15. <https://doi.org/10.1002/cpe.6989>
- Adane, K., Beyene, B., & Abebe, M. (2024). ML and DL-based Phishing Website Detection: The Effects of Varied Size Datasets and Informative Feature Selection Techniques. *Journal of Artificial Intelligence and Technology*, 4(1), 18–30. <https://doi.org/10.37965/jait.2023.0269>
- Airehrour, D., Nair, N. V., & Madanian, S. (2018). Social engineering attacks and countermeasures in the New Zealand Banking System: Advancing a user-reflective mitigation model. *Information (Switzerland)*. <https://doi.org/10.3390/info9050110>
- Akande, O. N., Gbenle, O., Abikoye, O. C., Jimoh, R. G., Akande, H. B., Balogun, A. O., & Fatokun, A. (2023). SMSPROTECT: An automatic smishing detection mobile application. *ICT Express*. <https://doi.org/10.1016/j.ict.2022.05.009>
- Al-Sarem, M., Saeed, F., Al-Mekhlafi, Z. G., Mohammed, B. A., Al-Hadhrami, T., Alshammari, M. T., Alreshidi, A., & Alshammari, T. S. (2021). An optimized stacking ensemble model for phishing websites detection. *Electronics (Switzerland)*. <https://doi.org/10.3390/electronics10111285>
- Aliza, H. Y., Nagary, K. A., Ahmed, E., Puspita, K. M., Rimi, K. A., Khater, A., & Faisal, F. (2022). A Comparative Analysis of SMS Spam Detection employing Machine Learning Methods. *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, 916–922. <https://doi.org/10.1109/ICCMC53470.2022.9754002>
- Alnemari, S., & Alshammari, M. (2023). Detecting Phishing Domains Using Machine Learning. *Applied Sciences (Switzerland)*, 13(8). <https://doi.org/10.3390/app13084649>
- Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., & Faheem, M. H. (2023). A Deep Learning-Based Phishing Detection System Using CNN, LSTM, and LSTM-CNN. *Electronics*, 12(1), 232. <https://doi.org/10.3390/electronics12010232>

- Amir Sjarif, N. N., Mohd Azmi, N. F., Chuprat, S., Sarkan, H. M., Yahya, Y., & Sam, S. M. (2019). SMS spam message detection using term frequency-inverse document frequency and random forest algorithm. *Procedia Computer Science*, *161*, 509–515. <https://doi.org/10.1016/j.procs.2019.11.150>
- Arab, M., & Sohrabi, M. K. (2017). Proposing a new clustering method to detect phishing websites. *Turkish Journal of Electrical Engineering and Computer Sciences*. <https://doi.org/10.3906/elk-1612-279>
- Azeem, S. (2020). *Customer behaviours and online banking in New Zealand*. Digitalnz.
- BioCatch. (2024). *BioCatch partners with Australian banks on launch of fraud and scams intelligence-sharing network*. Prnewswire.
- Bird, S., Klein, E., & Loper, E. (2013). *NLTK - Natural Language Toolkit*. [Http://Www.Nltk.Org/](http://www.nltk.org/).
- Chen, J., Li, Q., Wang, H., & Deng, M. (2020). A machine learning ensemble approach based on random forest and radial basis function neural network for risk evaluation of regional flood disaster: A case study of the yangtze river delta, China. *International Journal of Environmental Research and Public Health*. <https://doi.org/10.3390/ijerph17010049>
- Chen, T., & Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. <https://doi.org/10.1145/2939672.2939785>
- Chen, T., & He, T. (2014). xgboost: Extreme Gradient Boosting. *R Lecture*.
- Delany, S. J., Buckley, M., & Greene, D. (2012). SMS spam filtering: Methods and data. In *Expert Systems with Applications*. <https://doi.org/10.1016/j.eswa.2012.02.053>
- Dharani, V., Hegde, D., & Mohana. (2023). Spam SMS (or) Email Detection and Classification using Machine Learning. *Proceedings - 5th International Conference on Smart Systems and Inventive Technology, ICSSIT 2023*. <https://doi.org/10.1109/ICSSIT55814.2023.10060908>
- Do, N. Q., Selamat, A., Krejcar, O., Yokoi, T., & Fujita, H. (2021). Phishing webpage classification via deep learning-based algorithms: An empirical study. *Applied Sciences (Switzerland)*. <https://doi.org/10.3390/app11199210>

- Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*. <https://doi.org/10.1016/j.ejor.2017.11.054>
- Gadde, S., Lakshmanarao, A., & Satyanarayana, S. (2021). SMS Spam Detection using Machine Learning and Deep Learning Techniques. *2021 7th International Conference on Advanced Computing and Communication Systems, ICACCS 2021*. <https://doi.org/10.1109/ICACCS51430.2021.9441783>
- Gandhi, C., Kumar Sarangi, P., Saxena, M., & Sahoo, A. K. (2023). SMS Spam Detection Using Deep Learning Techniques: A Comparative Analysis of DNN Vs LSTM Vs Bi-LSTM. *Proceedings of International Conference on Computational Intelligence and Sustainable Engineering Solution, CISES 2023*. <https://doi.org/10.1109/CISES58720.2023.10183634>
- Ghourabi, A. (2021). SM-Detector: A security model based on BERT to detect SMiShing messages in mobile environments. *Concurrency and Computation: Practice and Experience*, 33(24). <https://doi.org/10.1002/cpe.6452>
- Goel, D., Ahmad, H., Jain, A. K., & Goel, N. K. (2024). *Machine Learning Driven Smishing Detection Framework for Mobile Security*.
- Gualberto, E. S., De Sousa, R. T., De Vieira, T. P. B., Da Costa, J. P. C. L., & Duque, C. G. (2020). From Feature Engineering and Topics Models to Enhanced Prediction Rates in Phishing Detection. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2020.2989126>
- Gupta, B. B., Gaurav, A., Attar, R. W., Arya, V., Alhomoud, A., & Chui, K. T. (2024). Optimized Phishing Detection with Recurrent Neural Network and Whale Optimizer Algorithm. *Computers, Materials & Continua*, 80(3), 4895–4916. <https://doi.org/10.32604/cmc.2024.050815>
- Hameed, S. M. (2021). Differential evolution detection models for SMS spam. *International Journal of Electrical and Computer Engineering*. <https://doi.org/10.11591/ijece.v11i1.pp596-601>
- Hapase, D. S., & Patil, L. V. (2024). Telecommunication fraud resilient framework for efficient and accurate detection of SMS phishing using artificial intelligence techniques. *Multimedia Tools and Applications*, 83(41), 89111–89133. <https://doi.org/10.1007/s11042-024-19020-2>

- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*.
<https://doi.org/10.1162/neco.1997.9.8.1735>
- Jacovi, A., Shalom, O. S., & Goldberg, Y. (2018). Understanding Convolutional Neural Networks for Text Classification. *EMNLP 2018 - 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Proceedings of the 1st Workshop*.
<https://doi.org/10.18653/v1/w18-5408>
- Jain, T., Garg, P., Chalil, N., Sinha, A., Verma, V. K., & Gupta, R. (2022). SMS Spam Classification Using Machine Learning Techniques. *Proceedings of the Confluence 2022 - 12th International Conference on Cloud Computing, Data Science and Engineering*.
<https://doi.org/10.1109/Confluence52989.2022.9734128>
- Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., & Alegre, E. (2023). Classifying spam emails using agglomerative hierarchical clustering and a topic-based approach. *Applied Soft Computing*. <https://doi.org/10.1016/j.asoc.2023.110226>
- Jefferson, Z. (2020). *Bank Data: SMOTE*. Analytics Vidhya.
- Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*. <https://doi.org/10.1186/s40537-019-0192-5>
- Källén, M., Sigvardsson, U., & Wrigstad, T. (2021). Jupyter Notebooks on GitHub: Characteristics and Code Clones. *Art, Science, and Engineering of Programming*.
<https://doi.org/10.22152/programming-journal.org/2021/5/15>
- Karhani, H. El, Jamal, R. Al, Samra, Y. B., Elhajj, I. H., & Kayssi, A. (2023). Phishing and Smishing Detection Using Machine Learning. *Proceedings of the 2023 IEEE International Conference on Cyber Security and Resilience, CSR 2023*.
<https://doi.org/10.1109/CSR57506.2023.10224954>
- Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.
- Lavanya, A., Sindhuja, S., Gaurav, L., & Ali, W. (2023). A Comprehensive Review of Data Visualization Tools: Features, Strengths, and Weaknesses. *International Journal of Computer Engineering in Research Trends*.
<https://doi.org/10.22362/ijcert/2023/v10/i01/v10i0102>

- Li, D.-C., Wang, S.-Y., Huang, K.-C., & Tsai, T.-I. (2022). Learning class-imbalanced data with region-impurity synthetic minority oversampling technique. *Information Sciences*, *607*, 1391–1407. <https://doi.org/10.1016/j.ins.2022.06.067>
- Li, D. C., Wang, S. Y., Huang, K. C., & Tsai, T. I. (2022). Learning class-imbalanced data with region-impurity synthetic minority oversampling technique. *Information Sciences*. <https://doi.org/10.1016/j.ins.2022.06.067>
- Logunova, I. (2022). *Random Forest Classifier: Basic Principles and Applications*. Serokell.Io.
- Mahmud, T., Prince, A. H., Ali, H., Hossain, M. S., & Andersson, K. (2024). *Enhancing Cybersecurity : Hybrid Deep Learning Approaches to Smishing Attack Detection*. 1–21.
- Mambina, I. S., Ndibwile, J. D., & Michael, K. F. (2022). Classifying Swahili Smishing Attacks for Mobile Money Users: A Machine-Learning Approach. *IEEE Access*, *10*, 83061–83074. <https://doi.org/10.1109/ACCESS.2022.3196464>
- Manikkannan, P. (2023). SMS Spam Detection using Machine Learning. *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*. <https://doi.org/10.55041/ijjsrem27463>
- Maqsood, U., Ur Rehman, S., Ali, T., Mahmood, K., Alsaedi, T., & Kundi, M. (2023). An Intelligent Framework Based on Deep Learning for SMS and e-mail Spam Detection. *Applied Computational Intelligence and Soft Computing*. <https://doi.org/10.1155/2023/6648970>
- McCombie, S. (2008). Trouble in Florida: The genesis of phishing attacks on Australian banks. *Proceedings of the 6th Australian Digital Forensics Conference*, 113–128. <https://doi.org/10.4225/75/57b2712140cbd>
- Mehare, H. Bin, Anilkumar, J. P., & Usmani, N. A. (2023). The Python Programming Language. In *A Guide to Applied Machine Learning for Biologists* (pp. 27–60). Springer International Publishing. https://doi.org/10.1007/978-3-031-22206-1_2
- Mehdipour, F., Babenkov, E., Hewage, U. H. W. A., & Aharari, A. (2023). Banking Fraud Identification and Prevention. *Proceedings - 27th International Conference on Circuits, Systems, Communications and Computers, CSCC 2023, July*, 71–76. <https://doi.org/10.1109/CSCC58962.2023.00019>
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word

- representations in vector space. *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*.
- Minh, N., & Nguyen, K. (2019). *Real-time fashion items classification using TensorflowJS and ZalandoMNIST dataset*. August.
- Mishra, S., & Soni, D. (2019). SMS Phishing and Mitigation Approaches. *2019 12th International Conference on Contemporary Computing, IC3 2019*.
<https://doi.org/10.1109/IC3.2019.8844920>
- Mishra, S., & Soni, D. (2022). Implementation of ‘Smishing Detector’: An Efficient Model for Smishing Detection Using Neural Network. *SN Computer Science*.
<https://doi.org/10.1007/s42979-022-01078-0>
- Mishra, S., & Soni, D. (2023). DSmishSMS-A System to Detect Smishing SMS. *Neural Computing and Applications*. <https://doi.org/10.1007/s00521-021-06305-y>
- Mohammed, N. A., Abed, M. H., & Albu-Salih, A. T. (2022). Convolutional neural network for color images classification. *Bulletin of Electrical Engineering and Informatics*.
<https://doi.org/10.11591/eei.v11i3.3730>
- Mughaid, A., AlZu’bi, S., Hnaif, A., Taamneh, S., Alnajjar, A., & Elsoud, E. A. (2022). An intelligent cyber security phishing detection system using deep learning techniques. *Cluster Computing*. <https://doi.org/10.1007/s10586-022-03604-4>
- Murali, S., & Swapna, T. R. (2019). An empirical evaluation of temporal convolutional network for offensive text classification. *International Journal of Innovative Technology and Exploring Engineering*.
- Muzigura, G., & Casmir, R. (2023). Evaluation of Measures Taken by Telecommunication Companies in Preventing Social Engineering Attacks in Tanzania. *European Journal of Theoretical and Applied Sciences*. [https://doi.org/10.59324/ejtas.2023.1\(4\).114](https://doi.org/10.59324/ejtas.2023.1(4).114)
- Nagy, N., Aljabri, M., Shaahid, A., Ahmed, A. A., Alnasser, F., Almakramy, L., Alhadab, M., & Alfaddagh, S. (2023). Phishing URLs Detection Using Sequential and Parallel ML Techniques: Comparative Analysis. *Sensors*, 23(7), 3467.
<https://doi.org/10.3390/s23073467>
- Nowitz, J. (2018). *A Modern Perspective on Phishing: An investigation into susceptibility to phishing attacks between mobile and desktop email clients*. 1–23.

- Ora, A. (2020). Spam Detection in Short Message Service Using Natural Language Processing and Machine Learning Techniques. *National College of Ireland, Pp.1-27*.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*.
- Prabhanjan Raja. (2022). *What is Google Colab?* Scaler Topics.
- Priezkalns, E. (2024a). *Consumer perceptions and attitudes towards SMS advertising: Recent evidence from New Zealand*. Commsrisk.
- Priezkalns, E. (2024b). *New Zealand Finds First SMS Blaster Bank Smishing Fraud*. Commsrisk.
- Prusty, S. R., Sainath, B., Jayasingh, S. K., & Mantri, J. K. (2022). SMS Fraud Detection Using Machine Learning. *Lecture Notes in Networks and Systems*. https://doi.org/10.1007/978-981-19-0901-6_52
- Rashid, J., Mahmood, T., Nisar, M. W., & Nazir, T. (2020). Phishing Detection Using Machine Learning Technique. *Proceedings - 2020 1st International Conference of Smart Systems and Emerging Technologies, SMART-TECH 2020*. <https://doi.org/10.1109/SMART-TECH49988.2020.00026>
- Rifat, N., Ahsan, M., Chowdhury, M., & Gomes, R. (2022). BERT Against Social Engineering Attack: Phishing Text Detection. *IEEE International Conference on Electro Information Technology*. <https://doi.org/10.1109/eIT53891.2022.9813922>
- Salman, M., Ikram, M., & Kaafar, M. A. (2022). An Empirical Analysis of SMS Scam Detection Systems. In *Proceedings of Oct* (Vol. 1, Issue 1). Association for Computing Machinery. <https://doi.org/10.48550/arXiv.2210.10451>
- Sharaff, A., Pathak, V., & Paul, S. S. (2023). Deep learning-based smishing message identification using regular expression feature generation. *Expert Systems*. <https://doi.org/10.1111/exsy.13153>
- Shinde, A., Shahra, E. Q., Basurra, S., Saeed, F., AlSewari, A. A., & Jabbar, W. A. (2024). SMS Scam Detection Application Based on Optical Character Recognition for Image Data Using Unsupervised and Deep Semi-Supervised Learning. *Sensors*, 24(18), 6084.

<https://doi.org/10.3390/s24186084>

- Snehkunj, R., Vachiyatwala, K., & Author, C. (2022). Data Analysis Using Pandas Library of Python. *Acta Scientific COMPUTER SCIENCES*.
- Srinivasarao, U., & Sharaff, A. (2023). Machine intelligence based hybrid classifier for spam detection and sentiment analysis of SMS messages. *Multimedia Tools and Applications*. <https://doi.org/10.1007/s11042-023-14641-5>
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*.
- Su, Y. (2020). Research on Website Phishing Detection Based on LSTM RNN. *Proceedings of 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2020*. <https://doi.org/10.1109/ITNEC48623.2020.9084799>
- Ulfath, R. E., Alqahtani, H., Hammoudeh, M., & Sarker, I. H. (2021). Hybrid CNN-GRU Framework with Integrated Pre-trained Language Transformer for SMS Phishing Detection. *ACM International Conference Proceeding Series*. <https://doi.org/10.1145/3508072.3508109>
- Valerian Chinedum, A., Bethran Chibuike, A., & A. Onyekachi, A. (2023). DETECTING ONLINE FAKE NEWS USING MACHINE LEARNING AND TOPOLOGICAL DATA ANALYSIS. *International Journal of Engineering Applied Sciences and Technology*. <https://doi.org/10.33564/ijeast.2023.v07i10.006>
- Wright, R. T., Johnson, S. L., & Kitchens, B. (2023). PHISHING SUSCEPTIBILITY IN CONTEXT: A MULTILEVEL INFORMATION PROCESSING PERSPECTIVE ON DECEPTION DETECTION. *MIS Quarterly: Management Information Systems*. <https://doi.org/10.25300/MISQ/2022/16625>
- Xu, X., Chen, W., & Sun, Y. (2019). Over-sampling algorithm for imbalanced data classification. *Journal of Systems Engineering and Electronics*. <https://doi.org/10.21629/JSEE.2019.06.12>
- Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A. K., & Almotairi, S. (2022). A Comparison of Pooling Methods for Convolutional Neural Networks. In *Applied Sciences (Switzerland)*. <https://doi.org/10.3390/app12178643>

- Zaini, N. S., Stiawan, D., Razak, M. F. A., Firdaus, A., Wan Din, W. I. S., Kasim, S., & Sutikno, T. (2020). Phishing detection system using machine learning classifiers. *Indonesian Journal of Electrical Engineering and Computer Science*, 17(3), 1165. <https://doi.org/10.11591/ijeecs.v17.i3.pp1165-1171>
- Zimba, A., Phiri, K., Kashale, C., & Phiri, M. (2024). A machine learning and natural language processing-based smishing detection model for mobile money transactions. *International Journal on Information Technologies and Security*, 16(3), 69–80. <https://doi.org/10.59035/UEEN6450>