A CYBER SECURITY-BASED CONVOLUTIONAL NEURAL NETWORK DETECTION MODEL FOR EMAIL PHISHING USING LONG SHORT-TERM MEMORY ALGORITHM

MURIITHI NICHOLAS MURIUKI

PA303/S/21210/23

A THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR AWARD OF THE DEGREE OF DOCTOR OF PHILOSOPHY IN INFORMATION TECHNOLOGY IN THE SCHOOL OF PURE AND APPLIED SCIENCES, KIRINYAGA UNIVERSITY

SEPTEMBER, 2026

Declaration

This Thesis is my original work and has not been presented for a degree or other award in any other University.

Signature: …… Date: ……..

Nicholas Muriuki Muriithi

PA303/S/21210/23

We confirm that the work reported in this thesis was carried out by the candidate under our supervision.

Signature: **Date:**

Dr. Ephantus Mwangi, PhD
Lecturer, School of Pure and Applied Sciences – Kirinyaga University

Signature: **Date:**

Dr. Kennedy Malanga, PhD
Senior lecturer, School of Pure and Applied Sciences – Karatina University

©

NICHOLAS MURIUKI MURIITHI

All rights reserved. No part of this project may be reproduced, stored in any retrieval system or transmitted in any form or by any means, electronically. Mechanically, by photocopying or otherwise, without prior written permission of the author or Kirinyaga University on that behalf

Dedication

I would like to dedicate this work to my family, my beloved wife, my friends, and my classmates, as they gave me encouragement, support, and inspiration in my academic life.

Acknowledgement

To begin with, I would like to express my gratitude to God as I am healthy and I have sufficient strength to complete this research work successfully. I also would not disregard those who supported me during the research process; without their help, this would not be possible.

Particular attention must be given to the parents and my wife, whose help in terms of finances I could count on throughout my whole project. I also wish to give a heartfelt appreciation to my supervisors, Dr. Ephantus Mwangi and Dr. Kennedy Malanga, who guided me and supported me and made the requisite information and knowledge available to me, so that my research could be successful.

Abstract

Phishing attacks have become more sophisticated, fraudsters pretending to be genuine companies by means of emails and messaging apps in order to obtain people's login information. Previous approaches relied on manually created features together with conventional machine learning techniques. These methods were unable to identify both small patterns in text and longer-term text trends. Moreover, a great many deep learning models only examine individual characters or words. This study had four primary objectives: (1) to examine and evaluate the current techniques for detecting phishing; (2) to develop a new model that combines a Convolutional Neural Network with a Long Short-Term Memory (CNN-LSTM); (3) to compare this model with standard machine learning methods; and (4) to use statistical tests to verify the results. The CNN was used to detect important small text patterns, while the LSTM analyzed the text in both the forward and backward directions, thus reducing the need for manually created features. The dataset came from Kaggle and consisted of 8,986 emails, half of which were phishing emails. The data was cleaned, split into words, and brought to the same length. It was divided so that 70% was used for training and 30% for testing (2,696 emails for testing). The model's performance was assessed using accuracy, precision, recall, F1-score, ROC-AUC, and a confusion matrix. Accuracy refers to the proportion of correct predictions out of the total number of predictions. The CNN-BiLSTM model reached an accuracy of 97.88%, a precision of 90.85%, and a ROC-AUC of 0.9798. It outperformed both the K-Nearest Neighbors and Random Forest methods by around 11 to 12 percentage points in the key areas and had fewer false positives and false negatives. The results were statistically significant ($p < 0.05$).However, these findings are from only one trial and the capability of this model with other data sets was not verified. The proposed CNN-LSTM model seems to be a very effective technique to detect phishing emails in the selected data set. However, factors such as scalability, speed, multilingual support, real-time detection capability, cloud connection, and resilience to novel attacks were not analyzed and need to be investigated further.

Table of Contents

Declaration ..... **i**

Dedication ..... **iii**

Acknowledgement ..... **iv**

Abstract ..... **v**

List of abbreviations and acronyms ..... **xii**

CHAPTER ONE **1**

INTRODUCTION **1**

1.1 Introduction **1**

1.2 Background **5**

1.3 Statement of Problem **10**

1.4 Purpose of the study **11**

1.5 Objectives **12**

1.5.2 Specific Objective **12**

1.6 Research Questions **13**

1.7 Scope **13**

Operational Definition of Terms **14**

CHAPTER TWO: **16**

LITERATURE REVIEW **16**

2.1 Introduction **16**

2.2 Theoretical Review **16**

2.2.1 Computational Learning Theory 16

2.2.2 Signal Detection Theory (SDT) 18

2.3 Overview of an Ensemble Learning Technique 20

2.4 Foundation of the CNN model in Phishing Detection 22

2.5. Empirical Review 22

2.5.1. Related studies on phishing email detection models 22

2.5.2. Metric for email phishing detection Techniques 91

2.5.3. Comparative analysis of the algorithms 92

2.6 Designing an Email Phishing Detection Model 96

2.7 Model Development 98

2.8 Model Validation 100

2.9 Research Gap 101

CHAPTER THREE 104

METHODOLOGY 104

3.1 Introduction 104

3.2 Research philosophy 104

3.3 Research paradigm 105

3.4 Research design 107

3.5 Model Development Procedure 109

3.6 Positive predictive value of the Model 120

3.7 Negative predictive value of the Model 121

3.8 Research population 121

3.8.1 Target population 122

3.8.2 Population Size 122

3.9 Ethical issues 123

3.10 Conclusion 124

CHAPTER FOUR 126

RESULTS AND DISCUSSION 126

4.1 Introduction 126

4.2 OBJECTIVE 1: Analysis of Existing techniques used in phishing attack detection. 127

4.2.2 DISCUSSION 131

4.2.2 CONCLUSION 143

4.3 Objective 2: Designing an efficient model for phishing detection based on the identified techniques. 143

4.3.1. Model using CNN-LSTM Technique Formulation 144

4.3.2 Logical Steps on Phishing Emails URLs detection model 146

Objective 4.11: Validating a CNN-LSTM Phishing Detection Model 167

4.11.1: Introduction 167

CHAPTER FIVE 190

SUMMARY, CONCLUSION AND RECOMMENDATIONS 190

5.1. Introduction 190

5.2. Summary 190

5.3 Conclusion 191

5.4. Recommendations 198

REFERENCES 200

APPENDICES 200

APPENDIX 1: UNIVERSITY INTRODUCTORY LETTER 200

APPENDIX 2: NACOSTI RESEARCH 201

APPENDIX 3: CODE SNIPPETS 202

APPENDIX 4: QUESTIONNAIRE 213

List of Tables

Table 2-1 Comparative analysis of the algorithm 94

Table 4-1 Selected Journals 127

Table 4-2 Techniques Distribution 130

Table 4-3 Training table distribution 154

Table 4-4 Confusion Matrix Table 156

Table 4-5 Model Metrics 158

Table 4-6 Model Filters 161

Table 4-7 Model comparison summary 163

Table 4-8 Correlation of variables 169

Table 4-9 Age Distribution 172

Table 4-10 Education Level 173

Table 4.11 Expertise 176

Table 4-12 Experience 178

Table 4-13 Prior use of detection Model 180

Table4-14 Summary Statistics 182

List of Figures

Figure 2-1 Ensemble architecture (Basit et al., 2020) 21

Figure 2-2 CNN Architecture 84

Figure 3.1 Model Development Procedure Flowchart 109

Figure 3.2 Dataset integration 111

Figure 3.3 Removing duplicates 112

Figure 3.4 Text Splitting 113

Figure 4.1 Algorithm Used 131

Figure 4.2 Model framework 144

Figure 4.3 Dataset Distribution 151

Figure 4.4 Training distribution 152

List of abbreviations and acronyms

3Q-Third Quarter

AI-Artificial intelligence

ANN-Artificial Neural Networks

APWG-Anti-Phishing Working Group

BERT-Bidirectional Encoder Representations from Transformer

CA-Communication Authority

CLT-Computational Learning Theory

CNN –Convolutional neural networks

DL-Deep Learning

DL-Deep Learning

DNN-Deep Neural Network

DNS-Domain Name System

DOS-Denial of Service

DT-Decision Tree

F1-F1-Score

FN-False Negative

FP-False Positive

FPR-False positive Rate

FTR-False True Rate

GloVe- Global Vector

HTML-Hypertext Markup Language

IM-Instant Messaging

IT -Information Technology

KNN-K-Nearest Neighbors

LSTM-Long short-Term Memory

ML-Machine learning

NLP-Natural language processing

NPV-Negative Predictive Value

PPV/NPV-Positive predictive value/False predictive value

PPV-Positive Predictive Value

PWCAC-Phishing Websites Classification Using Association Classification

PWCAC-Phishing Websites Classification Using Association Classification

RCNN-Region Based Convolutional Neural Networks

RF-Random forest

RNN-Recurrent Neural Network

ROC-AUC -Receiver operating characteristics -Area under the Curve

SDT-Signal Detection Theory

SPSS-Statistical Package for the Social Sciences

SVM-Support Vector Machine

TN-True Negative

TPR/FPR-True positive Rate/False positive Rate

TPR-True positive Rate

TP-True positive

URL-Universal Resource Locator

CHAPTER ONE

INTRODUCTION

1.1 Introduction

The rapid progress in information and communication technologies has changed how people, companies, and governments share information and communicate with each other. Among the different ways to communicate, email is still one of the most commonly used methods because it is easy to use, widely available, and inexpensive. However, the more people rely on email, the more chances cybercriminals have to trick users through phishing attacks. Phishing is a type of cybercrime in which hackers impersonate legitimate organizations to obtain sensitive information, such as login details, financial data, and personal information (Basit et al., 2021). As more organizations move toward digital transformation, phishing has become one of the most common and harmful cybersecurity threats worldwide.

Cybersecurity involves safeguarding internet-connected systems, encompassing data, software, and hardware, from cyber threats posed by malicious actors. It is essential to secure computer systems and information technology, as well as to compel businesses and other entities to defend their data and systems against cyber threats. Examples of potential cyberattacks include viruses, phishing, Trojan horses, worms, Denial-of-Service (DoS) attacks, unauthorized access (such as stealing private information or intellectual property), and control system attacks (Srinivas & Kumar, 2019).

In recent years, phishing attacks have become more advanced and have happened more often. Modern phishing efforts use methods like hiding real URLs, using short links, fake domains, social engineering, and multimedia tricks to get past basic security systems (Verma & Hossain, 2020).

Attackers often fake trustworthy links and use persuasive language to persuade people into doing things that put their security at risk. As a result, phishing has caused big financial losses, data leaks, identity theft, and damage to the reputation of individuals and businesses globally (Ahmed et al., 2022).

To deal with phishing, researchers and cybersecurity experts have created various ways to detect it, such as blacklists, rule-based filters, and machine learning algorithms like Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), and K-Nearest Neighbors (KNN) (Raza et al., 2021; Ahmed et al., 2022).

These methods have had some success in identifying known phishing patterns and harmful text and URLs. For example, Random Forest models have shown detection accuracy over 97% in phishing tasks, while Support Vector Machines have reached above 95% accuracy in some datasets (Alsariera et al., 2020; Safi & Singh, 2023). Even though these techniques have worked, they mainly rely on manually created features and known patterns, which makes them less effective against new and changing phishing methods (Taloba & Ismail, 2019).

A major problem with traditional phishing detection methods, like blacklists, whitelists, rule-based systems, signature-based approaches, and standard machine learning models such as Decision Trees, Naïve Bayes, Support Vector Machines, and Random Forests, was their failure to properly understand the meaning and order of words in emails and URLs. These methods mainly depended on fixed rules, known attack patterns, or manually created features, which made them less able to detect advanced, hidden, or new phishing attacks. As a result, their accuracy dropped when dealing

with phishing tactics that used different contexts and language styles (Aljofey et al., 2020).

Heuristics, and blacklisting/whitelisting systems focused on manually created features like URL length, domain age, word characteristics, and special characters. Even though these features can help find known phishing methods, they were not good at catching advanced attempts that use hidden URLs, shortened links, fake domains, and manipulate the context of emails (Sahingoz et al., 2019). Also, phishing attacks have changed quickly, so detection systems need to be able to adjust to new threats without a lot of manual help.

Recent advances in artificial intelligence and deep learning have given new possibilities for improving phishing detection. Deep learning can automatically learn features from raw data and find complex patterns that are hard to spot with traditional methods. Among these, Convolutional Neural Networks (CNNs) have proven very useful in identifying local and structural features from text, URLs, and cybersecurity data (Hussain et al., 2023). CNNs can find suspicious language, harmful characters, and inconsistent structures linked to phishing attacks, reducing the need for manual feature creation.

While CNNs are good at finding local patterns, they are not as effective at understanding long-term context in sequences. Long Short-Term Memory (LSTM) networks were created to address this by learning how events are connected over time. Long Short-Term Memory (LSTM) networks take this further by processing data both forward and backward, giving a better understanding of context in text sequences (Kavya & Sumathi, 2024). This is especially useful in phishing detection, where meaning and purpose often depend on the context of words, phrases, and links.

Research has shown that hybrid deep learning models combining CNN and LSTM can perform better than models that use only machine learning or deep learning alone (Altwaijry et al., 2024).In these models, CNN layers find local features in emails and URLs, while LSTM layers understand the context and sequence. Together, they can detect both known and new phishing attacks. These hybrid models have shown high accuracy, better ability to handle new situations, and greater adaptability to changing phishing techniques (Kavya & Sumathi, 2024).

Even with the progress in phishing detection, existing solutions still have issues like false positives, difficulty in keeping up with new attack methods, and the need for real-time performance. Cybercriminals keep modifying their tactics to stay hidden, so it is important to create smart models that can learn and adapt quickly. There is still a big need for more advanced systems that can automatically extract complex information, understand context, and maintain high accuracy in always-changing cybersecurity environments (Aljofey et al., 2020).

This study aims to tackle these challenges by developing and testing a hybrid model combining Convolutional Neural Networks and Long Short-Term Memory (CNN-LSTM) for phishing detection. The model uses the ability of CNNs to extract features and the context understanding of LSTM networks to identify phishing emails and harmful links better. By automatically learning both structural and sequential information from email data, the model should improve detection accuracy, lower false alarms, and better protect against evolving phishing attacks. The results of this study are expected to help cybersecurity research and practice by providing a strong, adaptable framework for protecting users and organizations from phishing threats (Aljofey et al., 2020).

1.2 Background

Phishing attacks have become one of the most prevalent and rapidly evolving forms of cybercrime worldwide. According to Piñeiro and Portillo (2022), there was a 59% increase in phishing attempts after the start of the COVID-19 pandemic. This growth is mainly because of the fast move to digital communication, remote work, and more use of online services. Another study in Latin America found that 21% of companies said social engineering attacks, especially phishing, were the fastest-growing cyber threat. Nearly half of the people surveyed pointed to the pandemic-related changes in how people use digital tools as the reason for this rise.

Phishing is still a major problem around the world, causing people to lose their login details, money, and sensitive data (Anti-Phishing Working Group [APWG], 2025; Verizon, 2025).It's also becoming more common in Africa and Kenya (Interpol,2025). To deal with this, different places have created laws and policies.

Kenya has the Computer Misuse and Cybercrimes Act (2018), the European Union has the NIS2 Directive (Directive (EU) 2022/2555, 2022), and the NIST Cybersecurity Framework 2.0 includes a Detect function (National Institute of Standards and Technology [NIST], 2024).These laws rely on good technical tools to find phishing attacks before they cause damage.

One promising tool is using machine learning to detect phishing (Safi & Singh, 2023).However, the success of this tool depends on how well the model is built and tested using measures like accuracy, precision, and recall. Accuracy on its own can be misleading when the data is not balanced (Chicco & Jurman, 2020; Vujović, 2021).

This study focused on creating, testing, and evaluating a phishing detection model. The goal was to provide useful technical information that can help develop better detection tools and improve cybersecurity awareness and protection, which aligns with the goals of these laws and policies.

Globally, phishing remains a big risk for money and security. The Anti-Phishing Working Group (APWG) reported 180,768 phishing websites in the first quarter of 2019, showing a steady increase compared to the previous quarters. These attacks have caused huge financial losses, estimated between $61 million and $3 billion each year in the United States alone (Aljofey et al., 2020). More recent reports from ThreatLabz show a 58.2% increase in phishing attacks between January and December 2023, with attackers focusing more on cloud services, financial websites, and government systems (Gajjar & Taherdoost, 2024). This shows that phishing has not only gotten bigger but also more complicated, moving from general email campaigns to more focused and clever attacks.

Looking at the world, cybercrime is spread unevenly but has a big effect everywhere. Countries like the United States, China, Russia, Brazil, Nigeria, and India have seen a lot of cyberattacks because of things like big digital economies, strong networks of cybercriminals, and widespread use of online financial systems. For example, India was among the top countries in cybercrime frequency between 2022 and 2024, with a big rise in attacks targeting digital payment systems and government platforms (Datta et al., 2020). These trends show that both developed and developing countries are becoming more vulnerable to phishing and other cyber threats.

In Africa, the situation is similar but has its own unique issues. Cybercrime has grown a lot along with rapid digital changes, especially in mobile money systems, e-government services, and online banking. Serianu, an African IT advisory firm, estimated that cybercrime cost African economies about $3.5 billion in 2017, with Kenya losing around $210 million a year and Nigeria about $649 million during the same time (Kshetri, 2019). Although these numbers are from the past, more recent cybersecurity reports suggest that the losses have kept going up because of more digital use and better tactics from attackers.

Cybersecurity problems in Kenya have become more serious in recent years. From July 2022 to June 2023, the National KE-CIRT/CC found over 855 million cyber threats aimed at Kenya's important information systems, placing the country among the top three targeted nations in the region, along with South Africa and Nigeria (Communications Authority of Kenya [CA], 2023).

Most of the threats were system attacks, which took advantage of weak points in old or not well-protected systems. About 90% of the threats found between July and September 2023 were this type, and the report says this is because there is not enough money spent on cybersecurity and many systems are still old (Communication authority of Kenya (CA), 2023).Social engineering is also a big problem, and the most affected areas were internet service providers, cloud service companies, government groups, and schools (Communication authority of Kenya (CA), 2023).
Phishing continues to be an issue: the KE-CIRT/CC said it was the main type of cyber threat in Kenya late in 2024, and there was an increase in phishing and DDoS attacks in areas that handle personal information, like the healthcare sector (Communication authority of Kenya (CA), 2025).

A big example of the damage from these attacks was in July 2023, when the “Anonymous Sudan” hacking group shut down more than 5,000 government services on Kenya’s e-Citizen platform.

Even though there was no confirmed data breach, the incident showed how vulnerable digital government systems can be and how cyberattacks can mess up important public services, reduce trust, and cause problems for the economy (Africa.com, 2024). This highlights the need for stronger and more intelligent cybersecurity measures in Kenya’s growing digital world.

Even though there are various ways to detect phishing, traditional methods like blacklists and rules have big drawbacks. These systems rely on fixed patterns and are not good at spotting new or hidden phishing attacks. As cybercriminals keep changing their tactics, these old-fashioned systems cannot adjust quickly, leading to more false negatives and lower detection rates.

To solve these issues, heuristic-based methods were introduced to help improve detection by looking at things like URL structure, domain behavior, and word patterns. While these methods improved accuracy and reduced false positives compared to blacklists, they still depend a lot on manually created features and struggle with new attack types (Zhu et al., 2020).

Recent improvements in artificial intelligence and deep learning have brought more flexible solutions for phishing detection. Specifically, Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks have become popular because they can automatically get features and understand sequences in text and URLs. CNNs work well with local patterns in URLs, like character structures, while LSTMs can learn relationships in both forward and backward

directions, helping to understand meaning better (Ozcan et al., 2023).

Although CNN, LSTM, hybrid CNN-LSTM, and transformer-based models have shown good results in detecting phishing, there are still some issues. Many of these models are trained and tested on static datasets that do not fully capture the variety of phishing attacks that happen in the real world. Also, transformer models like BERT and RoBERTa require a lot of computational power, making them less practical for use in environments with limited resources (Ahmed et al., 2022).

Because of this, this study aims to create, build, and test a phishing detection model that uses a hybrid CNN-LSTM architecture. The goal was to see how well it can find phishing attacks while reducing false positives and adapting to different phishing patterns. Unlike other studies that mainly propose models, this research focuses on practical testing to see how well the CNN-LSTM approach works in real-world situations.

The main contribution of this study is to show how hybrid deep learning models can be effective for phishing detection and to support the development of stronger, more flexible, and scalable cybersecurity solutions that can tackle the changing nature of phishing threats in today's digital world.

KNN tends to show moderate performance compared to other classical machine learning algorithms, including Logistic Regression, Naïve Bayes, Decision Trees, and SVM on large-scale phishing data with lower generalization. The algorithms, such as Random Forest and XGBoost, can (or should) interactively combine features and address class imbalance, but KNN requires significant manual feature preprocessing and hyperparameter optimization (e.g., k, distance metric,

and feature scaling method).

1.3 Statement of Problem

Phishing attacks have become more sophisticated and dangerous. The attackers pretend to be reliable companies and use fake URLs and emails that look real. They also often change or shorten the text in the email to trick people into providing them with their private information (Bu et al., 2019; Zien et al., 2023). If we do not take action against these attacks, they can pose an important threat to people and organizations. People may lose their identity, money or have their accounts taken over.

Organizations may experience data leaks, fake transactions, problems with their operations, fines and loss of customers' trust. Because the attackers are finding new ways to perform these attacks faster than the traditional security tools can keep up, phishing attacks are becoming more successful and causing more damage to the victims. Therefore, there is a need for better methods of detecting these attacks, which is why this research is necessary.

Traditional ways to spot phishing included using Support Vector Machines (SVMs) and Decision Trees, which were created based on features that people made by hand. These methods are time-consuming to create and can be easily tricked by attackers who know what features are being checked by the system.

While most studies on phishing focus on the words and letters within a URL, few consider the meaning of field names, subdomains and query strings.

For example, Bu and Cho (2021) employed a character-level convolutional autoencoder to learn

the characteristics of URLs and emails. Roy et al. (2022) utilized deep learning models such as LSTM and GRU to classify phishing URLs based on the order of their characters. However, since these methods do not consider the meaning of the text, recent work such as Su and Su (2023) has suggested employing methods that consider the meaning of the text to detect more advanced phishing attacks.

Another way that hackers trick people into giving them their information is through phishing scams. These scams are even more difficult to spot than others, as hackers often use tricks like hiding URLs in messages sent to people via messaging apps or using Business Email Compromise (BEC) scams (Atlam & Oluwatimilehin, 2023; Jain & Gupta, 2019).

One of the ways in which attackers can bypass the security filters on emails is by using URL hiding within messaging apps (Jain & Gupta, 2019). These types of attacks will continue to slip through the defenses of most email users and businesses until tools are developed that can understand the meaning and setting of the links within these emails. This is the problem that this research aims to solve.

1.4 Purpose of the study

Phishing attacks are becoming more advanced, which is causing serious problems for people and businesses around the world. These modern attacks use tricky website addresses, fake domains, manipulation of people's behavior, and content that changes on the fly, making it hard for usual security tools to catch them. Current methods for spotting phishing attempts often miss new threats or wrongly flag harmless content, leading to mistakes and greater risks for cybersecurity. Because

of this, there is a need for better and more flexible ways to detect phishing that can spot harmful links and emails more accurately without making too many errors.

This study looks into creating a better system for detecting phishing in email messages. It aims to fix the problems with current detection tools by finding new ways to improve accuracy, cut down on false alarms, and better recognize new types of phishing attacks. The research hopes to help build smarter and more dependable tools that can better protect people and organizations from cyber threats.

The study also wants to test how well the new detection system works using standard evaluation methods and see if it can reliably and quickly spot phishing attempts in today's digital world. By tackling the challenges posed by constantly changing phishing tactics, the research aims to help move forward both cyber security studies and real-world practices**.**

1.5 Objectives

1.5.1 General Objective

To develop a Cyber Security-Based Convolution Neural Network Detection for detecting phishing URLs and text in emails.

1.5.2 Specific Objective

By the end of the study, the researcher should be able to:

1. Analyze the existing phishing attacks detection techniques to identify techniques /technology used, strengths, weaknesses, and security loopholes.

2. Design an efficient Phishing detection model on the identified techniques.
3. Develop an Email phishing model for detecting phishing attacks.
4. Validate the performance and effectiveness of the developed phishing detection model.

1.6 Research Questions

1. What are the existing techniques currently used in phishing attack detection, and how effective are they in identifying different types of phishing attacks?
2. How can an efficient model for phishing detection be designed by incorporating the Strengths of existing techniques while addressing their limitations?
3. What are the key components required to develop a functional phishing detection Cyber Security-Based Convolution Neural Network Detection model, and how can these components be integrated to improve detection accuracy and efficiency?
4. How can the performance and effectiveness of the developed phishing detection model be Validated, and what metrics can be used to assess its success in real-world scenarios?

1.7 Scope

The CNN-LSTM hybrid model improves email phishing detection by fusing LSTM's proficiency in capturing contextual and sequential information in email content with CNN's capacity to extract structural data (such as questionable patterns in URLs). By spotting both local irregularities and more general phishing tendencies, this integration increases detection accuracy (Verma & Hossain, 2020).

With a focus on automated feature extraction and sequence analysis, the model was trained and validated on a variety of datasets to guarantee its efficacy in detecting both established and new

phishing techniques (Verma & Hossain, 2020; Sahingoz et al., 2019). In addition to testing the model's adaptation to changing threats and integration into pre-existing cybersecurity frameworks, the scope also involves evaluating the model's real-time detection capabilities for deployment in platforms such as email systems, web browsers, and security gateways (Basit et al., 2021).

The study intended to provide technical insights that supported future developments in phishing detection technologies and enhance cybersecurity awareness and protection by concentrating solely on model design, implementation, and evaluation using metrics like accuracy, precision, and recall. The study aligned with the Computer Misuse and Cybercrimes Act (2018), the EU's NIS2 directive, and international efforts of the US, Cybersecurity framework (NIST), since phishing has been an international threat where all continents must ensure it is controlled.

Operational Definition of Terms

Convolutional Neural Networks- A subtype of deep learning methods, convolutional neural networks (CNNs) are mostly used for computer vision tasks, including image classification and recognition. Because convolutional layers are used to automatically learn spatial hierarchies of data, they are particularly helpful for applications like object detection, facial recognition, and video analysis.

Cybersecurity- The practice of safeguarding networks, systems, and data from cyberattacks, illegal access, damage, and theft is known as cybersecurity.

Cyber Threat- Any malicious act or behavior intended to compromise the availability, confidentiality, or integrity of data, networks, or systems is considered a cyber threat.

Instant Message - Instant Messaging (IM) is a form of online communication that enables real-time text- based message sending and receiving between individuals or groups.

Long Short-Term Memory- A recurrent neural network (RNN) type called LSTM was created to get around the drawbacks of conventional RNNs, particularly when handling long-term dependencies in sequential input.

Model- It's a trained computational model that learns patterns from email data to detect and classify legitimate and malicious emails.

Phishing- Phishing is a kind of cyberattack in which criminals pose as trustworthy organizations or persons in an attempt to trick victims into divulging private information, such as credit card numbers, login credentials, bank account numbers, or personal data.

Random Forest- In machine learning, Random Forest is a well-liked and effective ensemble learning technique that can be applied to both regression and classification problems.

URL- The key tool for identifying and locating resources on the internet.

CHAPTER TWO:

LITERATURE REVIEW

2.1 Introduction

This chapter reviews several hypotheses that help us understand how threats affect users. The study will concentrate on two theories, which include computational learning theory and signal detection theory, to describe how algorithms function in resolving issues that impact the URL. Computational learning theory and signal detection theory inform research on enhancing the robustness of phishing detection models by ensuring that algorithms not only learn effectively from data but also make consistent and reliable decisions when confronted with deceptive and obfuscated email text.

The study focuses on an empirical review of objectives derived from other algorithms used by researchers. The research focused on reviewing how other algorithms mitigate malicious threats that affect operations.

2.2 Theoretical Review

2.2.1 Computational Learning Theory

Computational Learning Theory (CLT) provides a rigorous basis for developing effective phishing detection models, including CNN–LSTM hybrids, by analyzing their ability to generalize to unseen samples from the training data. CLT highlights that, to reduce the discrepancy between empirical risk and real risk and avoid overfitting, deep architectures, which have many parameters, require a substantial amount of varied data (Altwaijry et al., 2024).

In this instance, the LSTM layers capture sequential relationships between tokens, progressively detecting subtle phishing cues. In contrast, the CNN layers serve as feature extractors, detecting lexical patterns in email content (such as phishing-specific terminology and suspicious URL fragments). Regularization strategies, including dropout, weight decay, and early stopping, are applied in accordance with CLT-based principles to control model complexity efficiently (Altwaijry et al., 2024).

According to recent hybrid CNN–LSTM research, early stopping lowers overfitting without sacrificing accuracy. Furthermore, empirical results consistently show outstanding detection performance: CNN-based models achieve 98–99% accuracy in email phishing challenges, demonstrating their remarkable generalization abilities (Kavya & Sumathi, 2024). Grounding this theory ensures that the CNN-LSTM techniques remain accurate and resilient, which is capable of adapting to evolving phishing threats in real-world deployments.

The defined theory benefits this research by providing a foundation for improving the CNN-LSTM phishing detection model, enhancing its generalization, and ensuring it performs effectively on both training data and unseen phishing email data (Kavya & Sumathi, 2024). Theory also informs optimal complexity control by preventing overfitting to the data and ensuring balanced capacity for phishing detection accuracy.

In this study, CLT was operationalized in the selection and preparation of the dataset, model architecture, training procedure, and generalization controls. First, the study used a labelled dataset containing both phishing and legitimate URL/email instances so that the model could learn discriminative patterns from examples representing the two classes. The dataset was divided into

training and testing subsets using a 70:30 split. The training data were used to learn the model parameters, whereas the unseen testing data were reserved for evaluating whether the learned patterns generalized to previously unseen instances. This separation directly operationalized the CLT principle of distinguishing empirical performance on training data from performance on unseen data.

Second, CLT informed the selection of the CNN-LSTM architecture because the research required a model capable of learning complex representations from phishing-related textual and URL information. The CNN component was used to automatically extract local patterns, including characteristic character sequences, words, URL fragments, and other local phishing indicators. The LSTM component was then used to capture sequential and contextual dependencies in the extracted representations. Therefore, the architecture was selected to provide sufficient representational capacity to learn complex phishing patterns while subsequently controlling that capacity through the training procedures.

2.2.2 Signal Detection Theory (SDT)

Signal Detection Theory (SDT) provided a useful framework for understanding and improving phishing detection algorithms by examining how decisions are made under uncertainty. According to SDT, the challenge of phishing detection involves distinguishing between two types of stimuli: phishing emails (signals) and authentic emails (noise). The decision-making process is represented by four outcomes: hits (phishing detected correctly), misses (phishing not discovered), false alarms (phishing flagged as legitimate emails), and accurate rejections (phishing accurately classed). Researchers can assess a system's sensitivity (its capacity to distinguish phishing from non-

phishing) and decision threshold (its bias toward detecting more or fewer phishing messages) using this methodology.

Recent research used SDT to enhance machine learning models for phishing detection. When evaluating URL-based phishing classifiers, Verma and Hossain (2020) employed SDT measures and demonstrated that adjusting the decision threshold could maximize detection depending on the relative cost of false positives. Similarly, SDT was used by AlEroud, Alsmadi, and Damiani (2022) to evaluate the difference between human and machine performance in phishing email recognition. This study demonstrated how SDT can be used to modify system thresholds to reduce false positives and false negatives. To ensure consistent performance in dynamic circumstances, SDT also offers a framework for adaptive systems that can adjust their assessment criteria as phishing strategies evolve (Zhou et al., 2019).

Essentially, by providing a statistical method to assess decision quality and enhance classifier resilience, Signal Detection Theory improves phishing detection. It is an essential part of both theoretical and applied cybersecurity research, as it enables cybersecurity models to balance accuracy and risk management. This theory has been useful for research by providing a statistical basis for measuring and enhancing the model's ability to classify legitimate and illegitimate emails, based on the trade-off between hits and false positives. By applying the theory, one can explicitly evaluate model sensitivity (true positive rate) and true negative rate, which aids in optimizing decision thresholds for different operational requirements. The theory helps researchers identify bias in the dataset, thereby providing balanced performance across different datasets and reflecting the diversity of attacks that may occur in reality.

In this study, SDT was applied directly by how the results were sorted and the way the CNN-LSTM model was checked. The classes were set up as phishing = 1 and legitimate = 0.The model gave a chance of phishing for each input, and a cutoff point was used to decide if it was phishing or not. The default cutoff was 0.5, so anything at or above this was called phishing, and anything below was called legitimate.

The four SDT results were measured using a confusion matrix. True positives were when phishing emails were correctly identified. False negatives were when phishing emails were wrongly labeled as legitimate. False positives were when real emails were wrongly marked as phishing. True negatives were when real emails were correctly labeled as legitimate. This helped measure the ideas of hits, misses, false alarms, and correct rejections in a clear way.

ROC-AUC was also used to see how well the model could tell phishing from legitimate emails when the cutoff point was changed. This is important for SDT because adjusting the cutoff affects how many phishing emails are found versus how many false alarms are made.

2.3 Overview of an Ensemble Learning Technique

Due to the cybersecurity risks posed by phishing emails, there is a growing trend in the use of ensemble learning methods to enhance phishing detection. Multiple machine learning-based studies on phishing detection were conducted by scholars, including the following. The scholar Taha (2021) used nonlinear regression to detect phishing sites. The system was implemented using support vector machines (SVMs) and the harmony search method.

The scholar used 20 characteristics and 11, 055 websites. The sensitivity of the detection method they proposed was 92.80%. The author Taha (2021) has shown a phishing identification approach using 17 natural language processing (NLP) characteristics and 209 word-vector characteristics. The Random Forest, with an accuracy of 89.9%, has reigned as the best among the hybrid approaches. The random forest method has become common to detect phishing attacks. Previous research has achieved acceptable accuracy with RF, suggesting that some researchers have employed the technique. The algorithms used are: Artificial Neural Network, K-Nearest Neighbors, Support Vector Machine, RF, Rotation Forest, and C4.5 (Basit et al., 2020). In a further study by Promoting Basit et al. (2020), the dataset was obtained from the University of California, Irvine (UCI) machine learning repository, containing 11055 URLs (6157 phishing sites and 4898 genuine sites). During the course of their investigations, they theorized the use of the following architecture, which involved the creation of a combination of random forest and artificial neural networks (ANN), the K Nearest Neighbor, and C4.5.

Ensemble architecture

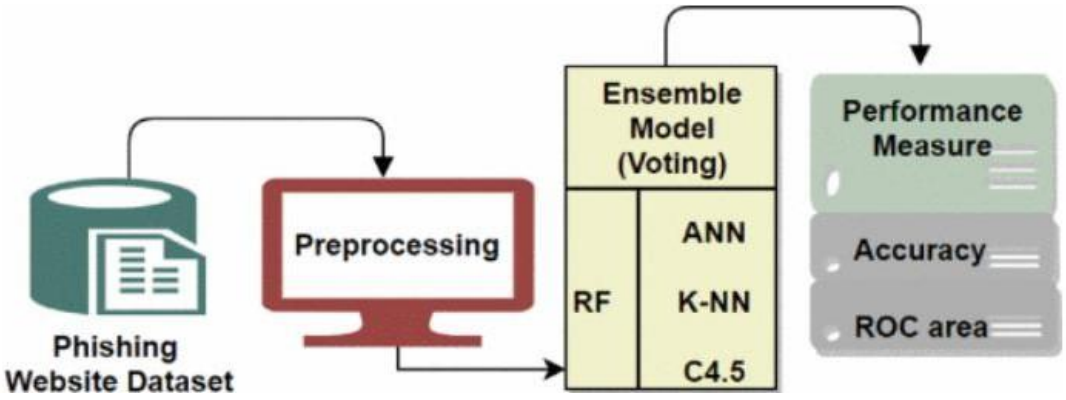

Figure 2-1 Ensemble architecture (Basit et al., 2020)

2.4 Foundation of the CNN model in Phishing Detection

Recent studies have shown that CNNs outperform RNNs in a wide range of sequence modeling tasks. Using both letters and words from the URL string, the researchers developed URLNet, which uses a CNN to learn URL embeddings in a jointly optimized framework (Hussain et al., 2023).

The academic and industrial communities have worked very hard to identify phishing websites. For instance, Google has implemented a blacklist that compiles a significant number of reported phishing websites to detect phishing attempts and uses the list in its own browser (Xiao, 2021).

2.5. Empirical Review

An empirical review of phishing emails comprised a methodical analysis of prior research and findings on the traits, efficacy, detection, and avoidance of phishing emails. Several studies have been conducted and are discussed below based on the criteria.

2.5.1. Related studies on phishing email detection models

i. Support Vector Machine

SVMs use kernels to project data into a higher-dimensional space, making it easier to separate the data with a hyperplane. This is especially useful when data is not clearly separated in its original form. Studies have shown that SVMs perform well in classifying complex phishing sets with overlapping patterns. However, they can be slow when handling large datasets or selecting the right kernel, which limits their use in real-time systems. One study reported an accuracy of 95.6% using PhishTank or public datasets, showing strong classification results that needed careful

feature tuning. It worked best with TF-IDF textual features and showed good performance in binary classification tasks (Raza et al., 2021).

A Support Vector Machine (SVM) created a hyperplane that helped classify the data. In two dimensions, this is a line, while in three, it is a plane. The method worked well with any number of dimensions. Raza et al. (2021) found that SVM can effectively detect spam even with small feature sizes, demonstrating good generalization. They used the SpamAssassin and PhishTank datasets, achieving 93% accuracy and outperforming KNN and Naive Bayes. However, it was computationally expensive and less easy to interpret than decision trees. Feature selection helped improve its performance.

Ahmed et al. (2022) proposed a method for detecting spam in social networks. They combined spam from different platforms, using 10,943 ham and 1,937 spam tweets from Twitter, as well as 9,285 ham and 1,338 spam posts. In TSD, 24.4% of tweets had different terms, while 75.6% had URLs. Of the ham tweets, 37.1% had only text, and 62.9% had both text and URLs. In FSD, 67.2% of spam postings were text-only, while 32.8% had links. Web links made up 95.1% of ham posts, with words making up the rest. They used the top 20 features from spam data on Twitter and Facebook and split the dataset into training and testing sets.

Zuraiq and Alkasassbeh (2019) developed a fast and accurate phishing detection method that combined Naive Bayes and SVM. Naive Bayes was used for initial web page detection, while SVM was used to reclassify questionable sites. They used 100 authentic and 100 phishing websites for training, and 600 phishing websites for testing. Phish Tank was the dataset source. The method

showed high accuracy and quick detection.

Singh et al. (2020) used a hybrid approach combining SVM, KNN, and logistic regression, achieving 98.0% accuracy with Alexa and PhishTank datasets. However, the model struggled with large-scale environments due to lower reliability in detecting legitimate and phishing instances.

Tawil et al. (2024) conducted an experiment to detect fraudulent emails using traditional machine learning. They tested accuracy in distinguishing genuine emails from phishing ones. The dataset included both email types, and preprocessing was performed to convert the textual content into numerical data for classification. They used three common models: Naive Bayes, Logistic Regression, and SVM. SVM performed best with an accuracy of 0.943, followed by Naive Bayes at 0.934 and Logistic Regression at 0.922.

According to Tawil et al. (2024) the strong performance of the SVM model was due to its ability to handle high-dimensional data and to find optimal decision boundaries that separate different classes. Emails often contain various textual features, such as words, phrases, and metadata, which are all part of the input features. SVM performed well in such a complex feature space by identifying the hyperplane that maximized the distance between different classes. This helped classify emails accurately, especially when the data contained similar patterns between real and phishing emails.

Naive Bayes was a probabilistic classifier based on Bayes' theorem. It is assumed that the features used for classification are independent, hence the term "naive." Despite this simplification, it has

proven very effective for text classification, including spam and phishing detection. Naive Bayes was popular because it's simple, computationally efficient, and works well with small training sets. In the study by Tawil et al. (2024) achieved an accuracy of 0.934, which was slightly lower than SVM, but still depicted how probabilistic models can successfully model email data patterns.

Logistic Regression was another common machine learning algorithm used for binary classification. In phishing detection, it was beneficial for determining whether an email was legitimate or fraudulent by estimating the probability that an input belonged to a specific class using a logistic function. It was interpretable and simple, making it useful for understanding the influence of specific features (Tawil et al., 2024). However, it may not perform as well as SVM or Naive Bayes when the data is complex or nonlinear. In the study, Logistic Regression achieved an accuracy of 0.922, which is lower than that of the other models but still effective at identifying fraudulent emails.

The research by Tawil et al. (2024) highlights the usefulness of traditional machine learning algorithms in detecting phishing. The high accuracy scores depicted that these models can effectively recognize patterns associated with fraudulent emails, such as suspicious keywords, email structure, misleading URLs, or unusual sender behavior. By using these features, machine learning models can be trained to tell the difference between normal and harmful email communications.

ii. Decision Tree (DT)

One of the most commonly used machine learning methods for tackling both classification and

regression tasks was the Decision Tree (DT). The algorithm worked by splitting a dataset into smaller parts, building a tree-like structure step by step. As explained by Salloum et al. (2022) the process involved recursively testing the relevance of features by looking at their quality levels. These quality levels indicate how well a feature can divide the data at each node of the tree. Two common measures for quality are the Gini Index and Entropy. The Gini Index reflected the probability that a randomly selected item is misclassified if it were assigned a label based on the distribution of labels in that subset. Lower Gini values mean a purer node (Salloum et al., 2022).

Entropy, on the other hand, measures the uncertainty in the data. It tells us how much information is needed to categorize an instance. It is closely related to the idea of information gain, which shows how much a feature reduces uncertainty during classification. During the learning phase of a decision tree, the feature with the highest information gain is typically chosen as the splitting point. Salloum et al. (2022) noted that entropy is directly tied to the information gain obtained during classification. This helps the algorithm to decide where to place nodes in the tree, whether they are internal or root nodes. The transparency of decision trees makes it easier for researchers and professionals to understand how data is being classified (Salloum et al., 2022).

Decision trees have been widely used in tasks which involved phishing and spam detection because they are simple and efficient. According to Raza et al. (2021) they applied the decision tree method in a three-tier design to classify spam and ham emails. Their model could detect spam in real time, making it highly useful for applications like email filtering and network security tools. The researchers found that the decision tree process is simpler than many other machine learning algorithms, which allows it to handle a high volume of emails quickly, a requirement for real-time

systems that must process thousands of messages.

Another advantage of decision trees, as stated by Raza et al. (2021) offered a clear and visual explanation of the classification process. System administrators and security analysts can easily understand how the decision tree arrived at a particular conclusion. The authors also mentioned that decision trees can detect recurring patterns in spam emails, such as suspicious domain names, unusual header information, or irregular use of Carbon Copy (Cc) and Blind Carbon Copy (Bcc) fields. These features are analyzed to classify emails as either spam or legitimate.

The data used for training and testing the model came from the UCI Machine Learning Repository or a personal phishing email dataset. Feature selection was essential in the decision tree process, as it helped identify the most effective features for classification. The model achieved an accuracy of about 96% in spam detection, showing its effectiveness in identifying phishing emails. However, there were some weaknesses. Poorly chosen features could lead to low interpretability and performance issues, unlike other machine learning techniques. To address these limitations, ensemble learning methods have been proposed. The Random Forest algorithm is a popular example. It combines multiple decision trees to improve classification accuracy and reduce overfitting. Alsariera et al. (2020 developed a smart phishing website detection model using the Random Forest method, which produced more accurate results (Raza et al., 2021).

Raza et al. (2021) evaluated the model using various measures such as the Receiver Operating Characteristic (ROC) curve, accuracy, and F-measure. These metrics are standard in cybersecurity research for assessing classification models. The ROC curve shows the relationship between the

true positive rate and the false positive rate, while the F-measure provides a balance between accuracy and recall.

In another study, Alsariedad et al. (2020) compared the Random Forest model to several other algorithms, including k-Nearest Neighbor (k-NN), Support Vector Machine (SVM), Artificial Neural Networks (ANN), Rotation Forest, C4.5, Classification and Regression Tree (CART), and Naive Bayes (NB). These algorithms were tested individually and in ensemble settings. The results showed that Random Forest performed the best, achieving an accuracy of 97.36%, an F-measure of 0.974, and an Area under the Curve (AUC) of 0.996, which indicates excellent classification performance.

Despite these promising results, the research had some limitations. The researchers mainly compared the performance of Random Forest with that of single classifiers like K-NN, SVM, and ANN. These individual models may not perform as well in certain cases compared to ensemble methods such as Random Forest (Alsareria et al., 2020). As a result, the comparison might not accurately reflect the performance of other ensemble-based techniques. To address these issues, the researchers introduced a new method called Phishing Websites Classification Using Association Classification (PWCAC). This method used association rules to classify websites as either genuine or phishing. By identifying relationships between different features in the dataset, this approach improves classification accuracy (Alsareria et al., 2020).

Another major contribution to decision tree research was made by Taloba & Ismail (2019). They developed a hybrid machine learning method known as Genetic Algorithm Decision Tree or

GADT. This approach combines genetic algorithms with decision tree learning. Genetic algorithms are optimization techniques inspired by natural selection and evolution. They help find the best parameter values that improve the machine learning model's performance.

In the GADT method, genetic algorithms were used to find the optimal value of the confidence factor, which controlled the pruning of the decision tree. Pruning branches, keeping tree depth in check, requiring a minimum number of samples, testing with cross-validation, and using group methods helped stop overfitting and make the model work better with new data. The researchers claim that combining genetic algorithms with decision trees can enhance the performance of text classification tasks such as spam detection (Taloba & Ismail, 2019).

Taloba & Ismail (2019) noted that one of the main challenges in spam and phishing detection is the large number of features in email datasets. Text is usually represented by thousands of words and patterns that may or may not be useful for classification. This high dimensionality can reduce classifier performance and make them more computationally demanding. The GADT model addresses this by optimizing parameters related to feature selection and pruning, leading to better classification results.

Despite its advantages, decision tree algorithms can be limited in contemporary phishing detection applications. They work well with simple and structured data but struggle with complex datasets that include advanced attack patterns. Current phishing attacks are highly sophisticated, involving obfuscated URLs, dynamically generated domains, and social engineering tactics that are difficult to detect using traditional feature engineering methods (Taloba & Ismail, 2019).

Moreover, decision trees are not very adaptable when new attack patterns emerge. Unlike deep learning models that can automatically learn complex features, decision trees rely heavily on manually engineered features. This makes them less effective at identifying new phishing attacks. This limitation highlights the need for more advanced machine learning models capable of handling dynamic cybersecurity threats (Taloba & Ismail, 2019).

Another study on decision trees was conducted by Ahmed et al. (2022) where the focus was on improving the accuracy of phishing website classification. The researchers used a publicly available dataset from the UCI Machine Learning Repository, which included 4898 phishing sites and 6157 legitimate sites.

To select the most relevant features for classification, the researchers employed feature selection techniques. Feature selection is a critical step as it reduces noise and removes irrelevant data. The selected features significantly improved the efficiency and accuracy of the decision tree model. The study's results showed a classification accuracy of 98.80 percent, demonstrating the effectiveness of decision trees in phishing website classification (Ahmed et al., 2022).

Although the study had high accuracy, the researchers suggested that it could be improved by combining decision trees with other machine learning techniques. Hybrid and ensemble methods have shown great potential in boosting classification performance in cybersecurity. For instance, decision trees can be paired with boosting algorithms such as AdaBoost or Gradient Boosting to detect complex attack patterns better, as noted by Ahmed et al. (2022).

Ahmed et al. (2022) also mentioned that new advancements in machine learning have led to the use of deep learning for phishing detection. Models like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been used to analyze both the text and structure of phishing emails. These models require less manual feature engineering since they can automatically extract meaningful patterns from large datasets. However, deep learning models usually demand substantial resources and a lot of training data.

Decision trees are also a favorable choice compared to deep learning models because they are easier to understand and simpler to implement. Explainability is critical in cybersecurity, as security analysts need to understand how models arrive at their decisions. Decision trees provide clear paths for decision-making that help analysts track how phishing emails are classified and identify suspicious patterns (Ahmed et al., 2022).

Ahmed et al. (2022) pointed out that decision trees are computationally inexpensive, making them suitable for real-time systems. Many organizations use decision tree-based models at email gateways and intrusion detection systems to block malicious messages before they reach users. This proactive approach helps prevent phishing attacks and protects sensitive company data.

In conclusion, decision trees are important in phishing and spam detection as a machine learning method. They are widely used in cybersecurity because of their clarity, fast classification abilities, and ease of implementation. Research has confirmed their effectiveness in identifying phishing websites and spam emails with high accuracy. However, they have limitations when dealing with complex and evolving attacks. To overcome these challenges, scholars have proposed hybrid and

ensemble methods that combine decision trees with other machine learning models. Future research could explore blending decision tree algorithms with deep learning frameworks and advanced feature extraction methods to combat modern phishing threats better, as outlined by Ahmed et al. (2022).

iii. K-Nearest Neighbour(KNN)

The K-Nearest Neighbors (KNN) algorithm is a type of non-parametric method that works by using existing data points to make predictions. It is called instance-based because it relies on the actual data examples rather than making assumptions about the data's underlying structure. This makes it effective even when the data has some noise or errors (Raza et al., 2021).

KNN works by looking at how close a new data point is to already labeled examples in a multi-dimensional space. This makes the method both easy to understand and versatile, as it can be used for both classification tasks, where it assigns categories, and regression tasks, where it predicts numerical values. This flexibility allows KNN to provide different types of results depending on what is needed for a specific application. However, despite these strengths, KNN is not widely used in large-scale studies because of several limitations. One major issue is that it is very sensitive to outliers in the data, and it can be computationally expensive when dealing with data that has many features (Raza et al., 2021).

These limitations are especially relevant in the context of phishing detection. Real-world datasets for phishing emails or URLs are often noisy, unbalanced, and may contain mislabeled examples. When people report phishing samples, the data can become noisy due to inconsistencies in labeling

or difficulties in extracting relevant features, which create uncertainty in the dataset. Despite these challenges, KNN has been found to be somewhat resilient to random noise. Its local decision-making process, which is based on the density of nearby data points rather than global assumptions, makes it less sensitive to small changes in the data distribution, especially when a suitable number of nearest neighbors (k) is chosen
(Slam et al., 2022).

Slam et al.(2022) found that using KNN on top of deep learning-generated features significantly improved phishing detection accuracy, especially in difficult cases that traditional models might struggle with. They showed that using KNN with a hybrid model of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks led to more accurate, context-based classifications. CNN layers are good at capturing spatial and local patterns in emails or URLs. In contrast, LSTM layers help understand the sequence of characters or words, allowing the model to capture both time-based and language-related dependencies better. By combining deep feature embeddings with the similarity-based approach of KNN, the model can make more detailed classifications, especially when phishing features are not clearly defined.

As Slam et al. (2022) noted the hybrid model was tested using a mix of UCI datasets and synthetic phishing data, achieving an accuracy of about 87%. However, they also pointed out that the system had trouble dealing with outliers and unevenly distributed data issues that KNN is known for. Since KNN directly uses all the training data to make classifications, the presence of outliers can distort the neighborhood structure, leading to incorrect distance measurements and misclassifications. This problem is particularly common in phishing email detection because there

are far more legitimate emails than phishing ones, which can bias the model towards predicting the majority class. This imbalance continues to be a challenge in the field.

i. Strengths and Suitability of KNN to Phishing Detection.

One of the main benefits of KNN is how simple and easy it is to understand. Unlike complex deep learning models that require a lot of training and fine-tuning, KNN is a lazy learner. This means it does not build a model during the training stage but instead waits until it needs to make a prediction. Because of this, KNN can quickly adapt to new data and be updated easily when new phishing examples are available. KNN is non-parametric, which means it does not make any assumptions about the data it works with. This is especially helpful in phishing detection, where the data might not follow a normal or linear pattern.

KNN is also good at capturing local features in the data, which is useful when small differences in language, structure, and writing style are important in telling apart real emails from phishing ones. For instance, things like how long a URL is, how often special characters appear, the randomness of a domain, and whether there are deceptive words can be quite different between legitimate and fake emails. KNN uses similarity measures like Euclidean or cosine distance to group these examples based on how they're related locally, rather than looking at overall trends.

KNN can also be used effectively in high-dimensional spaces when dealing with text-based phishing problems, especially when using techniques like Principal Component Analysis (PCA) or t-Distributed Stochastic Neighbor embedding (t-SNE) to simplify the data. Combining KNN with other models, such as Random Forests, Support Vector Machines (SVMs), or Gradient

Boosting, can lead to more reliable and consistent results. These mixed systems can benefit from KNN's ability to focus on local patterns and the broader predictive power of the other models. Even though KNN is useful, it does not always perform perfectly in real-world situations.

KNN has a number of performance drawbacks, even though the idea is a conceptually simple one. To begin with, it is linearly proportional to the size of the dataset. As all samples used in training should be stored and matched during the prediction time, it makes the algorithm prohibitively expensive when massive phishing datasets, involving hundreds of thousands of samples, are to be used. This high computational complexity, which is commonly represented as $O(n \times d)$, where n is the count of samples and d is the count of dimensions, precludes its scaling in the production-scale phishing detection systems.

Second, the vulnerability of KNN to outliers is a major disadvantage. Outliers may have a disproportionate effect on distance measurement, particularly in a lower-dimensional feature space. In phishing, where a genuine message can sometimes take on the characteristics of phishing (e.g., marketing emails that contain links embedded in them), such outliers can confuse the algorithm into false positive classifications. The issue is more critical in skewed data sets where phishing samples represent a small percentage of all the data, which is usually less than 10 percent. With this, KNN would be biased to the majority class and would perform well overall, but with low recall of phishing examples. This problem of imbalance is specifically worrying in security applications, where false negatives (phishing mail that has been missed) may result in serious security violations.

Third, the use of feature scaling and distance metric may have a major effect on the performance

of KNN. Due to the fact that the algorithm is purely based on the distance calculations, measurements of features across varying scales may skew the outcome. On the example of URL length, it might prevail over other text characteristics unless data are normalized or standardized. Equally, the decision of whether to use Euclidean, Manhattan, or cosine distance measures influences the neighborhood building. Cosine similarity is sometimes a superior method in phishing detection tasks that use textual or semantic embeddings than Euclidean distance because the former relies on directional similarity, compared to magnitude differences.

The other disadvantage is the curse of dimensionality. The more features one has, the less sense the concept of distance makes, and all points will look equally close to each other. This phenomenon impairs the discriminative capability of KNN in high-dimensional text-based representations, including TF-IDF and word embeddings. Thus, feature selection or dimensionality reduction is a critical step that should be performed before using KNN on phishing data.

ii. Deep Learning Architectures Integration.

In recent years, researchers have attempted to address these difficulties by combining KNN with deep learning. The logic behind such hybridization is that deep neural networks can automatically learn compact, discriminative feature representations, and KNN can classify at a fine-grained level on these learned representations. One of the examples of such synergy is the CNN-LSTM-KNN architecture, offered by Slam et al. (2022). Layers CNN identify local syntactic patterns in email text and LSTM layers identify long-term dependencies between the sequences of words. This information is fed into a KNN classifier, which makes the final decision based on the nearness of embedded representations.

This kind of integration enables KNN to be performed in a low-dimensional but semantically-enriched feature space, which effectively alleviates problems of noise and outliers. It also offers an explainability mechanism (as the classification of the KNN method relies on the closest examples, it is possible to trace decisions back to the particular training samples, which will help to improve interpretability). Nevertheless, the hybrid model continues to have certain weaknesses associated with KNN, including computational complexity and bias in the dataset.
Other studies have also indicated the same (Raza et al., 2021).

To illustrate this point, Raza et al. (2021) indicated that although KNN-based hybrid models can achieve a higher classification accuracy by 5-10 percent than the standalone neural networks, they are also expensive in terms of computational cost, which is limiting to real-time phishing detection. As a solution, the algorithm, approximate nearest neighbor (ANN), and optimized data structure, KD-trees or Ball-trees, have been used to provide faster distance calculations. Through these methods, time complexity can be defined as a logarithmic rather than linear value for a moderately sized dataset, KNN is more implementable in a security monitoring system.

iii. Solving the Imbalanced Dataset Problem.

Class imbalance is the question that still exists as one of the most severe problems of KNN in phishing detection. Phishing messages are comparatively few in number relative to legit ones, so the algorithm is more likely to recognize a phishing message as a legitimate one, and thus, the phishing messages are not well recalled. Several resampling approaches have been investigated to overcome this, such as Synthetic Minority Over-sampling Technique (SMOTE), Adaptive

Synthetic Sampling (ADASYN), and Near Miss under-sampling. These techniques provide artificial balancing of the data set by creating artificial phishing samples or randomly sampling out the majority samples, thus leveling out the data classes prior to the training of KNN.

Further to being resampled, the cost-sensitive KNN methods impose heavier misclassification fines on phishing examples. The algorithm can give preference to the correct detection of phishing samples without significantly reducing the overall accuracy by giving the contribution of minority-class neighbors a heavier weight. The ensemble learning models that integrate two or more KNN classifiers that were trained upon various balanced subsets have also proven to be useful in reducing problems of imbalances.

iv. Comparative Analysis and Research Perspective.

KNN tends to show moderate performance compared to other classical machine learning algorithms, including Logistic Regression, Naïve Bayes, Decision Trees, and SVM on large-scale phishing data with lower generalization. The algorithms, such as Random Forest and XGBoost, can (or should) interactively combine features and address class imbalance, but KNN requires significant manual feature preprocessing and hyperparameter optimization (e.g., k, distance metric, and feature scaling method).

However, KNN is useful as a baseline model and a support classifier in ensemble and hybrid deep learning models. The fact that it can be interpreted and make local decisions on its own makes it especially appropriate to post-classification analysis, where model predictions require interpretation or comparison with known phishing patterns. In addition, phishing techniques

change swiftly, and so the models that can update the new patterns quickly without training are beneficial. The lazy learning paradigm of KNN enables it to acquire the new samples at the time of their appearance, which is flexible in dynamic threatening situations.

It is assumed that future studies will be devoted to the improvement of KNN in terms of its scalability and strength with the help of algorithmic optimization and hybrid approaches. The neighborhood representation using graphs, where samples are modeled as nodes interrelated by similarity edges, might be improved in the case of graph-based learning, which can help to avoid sensitivity to outliers. Furthermore, similarity searches with embedding methods, like FAISS (Facebook AI Similarity Search) or HNSW (Hierarchical Navigable Small World graphs), can help execute nearest neighbor queries in high-dimensional space dramatically faster, which can now enable real-time phishing detection with variants of KNN.

iv. Random Forest

Phishing has turned out to be one of the most prevalent cyberattacks on individuals, organizations, and government institutions in the world. Malicious emails and malicious URLs are frequently employed to tempt unsuspecting users to provide sensitive information such as passwords, credit card details, and personal information. As a result, machine learning (ML) and deep learning approaches have more often been employed by researchers to detect a phishing attack automatically. Random Forest (RF), k-Nearest Neighbor (KNN), Decision Trees (DT), and Support Vector Machines (SVM) are the most popular ones and are included in the conventional algorithms of machine learning. Later on, more modern versions of deep learning architecture (Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (LSTM))

have been proposed to improve detection performance. Such articles as Safi and Singh (2023) and Bu and Cho (2021) come in handy when studying the advantages and disadvantages of such techniques.

Safi & Singh (2023) reported good results with the Random Forest classifier in identifying phishing URLs. The researchers used a few machine learning algorithms to measure the phishing detection performance based on the dataset gathered on the Kaggle platform. The data used comprised 11,504 URLs that were either phishing or legitimate. With such a dataset, the random forest model gave an accuracy rate of about 97%. Random Forest is an ensemble-learning algorithm that does not train a single decision tree, but rather builds a system of decision trees and will provide the final prediction in the form of the majority vote of the individual decision trees. This is a method of minimizing the overfitting that is normally offered by individual decision tree models. The large accuracy of the study indicates that the Random Forest is a good model when used to detect patterns that can be used to differentiate between a phishing and a legitimate URL.

Nevertheless, although the accuracy outcome is quite striking, some methodological constraints are also mentioned in the study. Among the key drawbacks, it is worth mentioning the fact that the authors of the research did not specify the hyperparameter configuration of the model. The choice of hyperparameters like the number of trees, the depth of the trees, and the method of feature selection can greatly influence the results of the Random Forest classifiers. It is hard to tell whether the reported accuracy is the best performance of the model as it is, without adequate hyperparameter tuning or trying various parameter settings. The research further failed to present a comparison with any of the past similar models with the same database. These comparisons are

significant in the research as they enable the reader to assess the extent to which the suggested method would indeed be better than the current methods (Safi & Singh, 2023).

The other constraint that is reported in the study is the one related to the dataset itself. The dataset that was acquired in the UCI Machine Learning Repository was used by Safi & Singh (2023) to make experiments, too. This data included 11,055 URLs, 6,157 of which were phishing URLs and 4,898 legitimate URLs. Three distinct experimental tests were done with this dataset by the researchers, who scored 97.36, 97.45, and 97.25 correct, respectively. These findings indicate the reproducibility of the Random Forest classifier when used in a series of experiments. The minor difference in the accuracy of the tests can be explained by the difference in the training and testing data split or by randomization of the model training.

Even though the performance outcomes are favorable, the research determines that there are a number of problems with the usage of publicly available datasets like those of the UCI repository. One of them is that open datasets are usually not normalized or processed through standardized procedures. Normalization is a crucial process in machine learning in the sense that it makes all features have an equal contribution towards the predictions of a model. The features with higher numeric values without normalization can overtake the learning process of the model, resulting in bias in predictions. Moreover, the data provided in the study did not contain the original URL, but rather it used extracted features. This implies that there is a possibility that valuable structural attributes of the URLs were lost in the preprocessing (Safi & Singh, 2023).

The other severe weakness is the fact that the study has used only one dataset to test the models. The results might not be generalizable due to the use of a single dataset, as the model might perform well on a particular dataset but not when working with real-world data that has varying characteristics. When analyzing models in cybersecurity research, it is a rule of thumb to test this model with the help of various datasets obtained in distinct places. This method is useful in making sure that the model will be flexible enough to accommodate various phishing techniques employed by criminals (Bu & Cho, 2021).

KNN, Decision Trees, and SVM are traditional machine learning algorithms that have been extensively applied in phishing detection studies owing to their simplicity and interpretability. The k-Nearest Neighbor method is used to classify new instances based on how similar the new instance is to its nearest training data. Although this technique can deliver positive outcomes in some instances, it is limited when used on big data sets since it involves the calculation of the distance between numerous data points. This may turn out to be computationally complex, especially when real-time detection is needed (Bu & Cho, 2021).

Phishing was another domain where Decision Trees was employed as it was easy to understand and interpret decision rules. The structure of the tree was easy to understand for security analysts on how the model arrives at the classification decisions. Nonetheless, Decision Trees tend to overfit when they are trained on complicated data. Overfitting happens when the model memorizes the patterns that it has learned on the training data, but does not apply to the unseen data. Though the ensemble algorithms like Random Forest can be used to mitigate this problem, the single decision tree model might still fail in highly dynamic phishing attacks (Bu & Cho, 2021).

Another conventional machine learning method that has been applied in the detection of phishing is Support Vector Machines. SVMs work especially well with high-dimensional data since they seek the best hyperplane over which to classify phishing and non-phishing cases. The SVMs, although effective, need a careful choice of the kernel functions and the parameter adjustment. Incorrect choice of parameters can greatly decrease the accuracy of classification. Additionally, SVM models are also computationally costly when large datasets are used with millions of samples (Bu & Cho, 2021).

A similar constraint that most of these traditional machine learning algorithms have in common is that they heavily depend on manual feature extraction. The feature extraction implies the selection of useful features of URLs or emails that can be used to draw a line between the phishing attempt and the legitimate one. Such features may include the length of the URL, suspicious characters in the URL, domain age, number of subdomains, and occurrence of certain keywords in email content. Although these features can offer valuable clues to classification, they have to be carefully crafted by hand with extensive domain knowledge, and not all the sophisticated trends of contemporary phishing attacks may be represented.

The other weakness is that classic machine learning frameworks tend to be unable to keep up with new and emerging attack strategies. Cybercriminals are constantly creating new ways of going around security systems, including creating email texts that are not static and employing advanced social engineering methods. Since classic models are based on pre-established characteristics, they

cannot easily identify phishing or attacks based on a pattern never witnessed before. This inability to change renders these models hard to work well in real-time phishing detection systems.

To overcome these problems, scientists have turned to deep learning solutions even more. According to Bu & Cho (2021), hybrid deep learning models such as CNN-LSTM may prove quite useful in the process of phishing detection. The structural patterns of input data are obtained with the assistance of the CNN part of the model, and sequential associations between the elements of input data are obtained with the help of the LSTM part of the model. This type of combination allows the model to analyze structural and semantic data of phishing emails, URLs, and links.

The property of CNNs was that it is highly efficient in local trends of information identification. In the case of phishing, CNNs can find the irregular patterns in URLs, email headings or email body texts automatically. Unlike traditional methods of machine learning, CNNs do not require using manual feature engineering because they also discover the features of interest automatically with the aid of the raw input data. This capability makes them quite suitable as far as analyzing a large and complex data.

Conversely, Long Short-Term Memory networks are networks that are intended to identify temporal, or sequential links in data. The sequence of words or characters in the spam mail can be the context to determine the spam or a genuine email. LSTM is bi-directional, and therefore, it enables the model to consider both forward and backward directions of the sequence to enable the model to capture a more detailed contextual information.

At the same time, both structural and semantic features can be determined with the help of CNN

and LSTM based on the hybrid model proposed by Bu and Cho (2021). Structural features may include features such as domain patterns, URL structure, and HTML elements, but semantic features include features of the meaning and setting of the email information. The latter duality feature allows the model to detect more complex phishing attacks, which may be overlooked by traditional machine learning techniques.

The other benefit of CNN-LSTM method is that it is suitable in the real-time phishing detection. New features of new data can be learned by the model, since the model can automatically extract and learn these features, it can learn new attack patterns quickly, and it does not necessarily have to learn new feature sets manually. This is needed in the current cybersecurity environments where the attackers are still evolving their ways to ensure they go undetected (Safi & Singh, 2023).

Besides, large-scale data processing models based on deep learning can execute large volumes of data at reduced efficiency compared to most traditional algorithms on current computing platforms such as GPUs and distributed computing systems. This scalability enables them to be used in large email systems and network monitoring systems, which necessitate the analysis of millions of messages per day (Safi & Singh, 2023).

Even though these are the privileges of deep learning models, challenges are also attached to them. One of the biggest challenges is the necessity to possess large amounts of labelled training data. The deep neural networks typically require thousands or even millions of samples to be trained in order to achieve a high degree of accuracy. This can be time-consuming and expensive in terms of collecting and labelling such datasets. Second, deep learning models are computationally

expensive to train, and are not always available in all research (Safi & Singh, 2023).

The other issue is that deep learning models are not interpretable. Deep neural networks are black boxes that are complicated, unlike decision trees or rule-based systems. This means that scientists and security experts might be unable to understand the way the model arrives at its conclusions. Explainability is often important in cybersecurity applications because it is important that the analysts explain why certain emails or URLs are malicious.

Altogether, the analysis of the articles by Safi & Singh (2023) and Bu & Cho (2021) reveals the shifting state of the research on the phishing detector. Random Forest and KNN, Decision Trees, and SVM are all conventional machine learning algorithms that continue to be useful when it comes to detecting phishing attacks, particularly in closed environments where features are clear-cut. These are not excessively complex models to work with and comprehend, and can be very precise when applied to well-prepared data sets.

However, these traditional methods also possess several disadvantages, including but not limited to the fact that they rely on manual feature engineering, cannot respond dynamically to the various threats, and are unable to handle complex patterns, which do manifest in the most recent phishing attacks. The CNN-LSTM and other deep learning techniques can be deemed as good solutions to these problems because they can automatically make structural and semantic features on the basis of raw data. They are more applicable in the real-time phishing detection system because they are able to process sequential information and learn to adjust to new patterns.

The future studies must focus on extending the available machine learning techniques to deep learning models to enable the development of hybrid models with the advantage of both techniques. In addition to that, the generalization of their models should be tested using multiple datasets that include different sources so that it can be viewed as a generalized model. The next important aspect that will be applicable to the improvement of phishing detection technology will be the quality of datasets, the feature extraction method, and the interpretability of the models.

v. Neural Networks

Deep learning models are considered one of the most powerful and advanced techniques for identifying phishing attempts and enhancing cybersecurity intelligence. This is due to their ability to uncover complex, nonlinear relationships from vast, seemingly random data. Unlike traditional machine learning methods that rely on manual feature selection and predefined statistical measures, neural networks automatically extract relevant features across multiple layers of abstraction. This allows the model to understand intricate patterns in the structure, behavior, and meaning of phishing data. This capability makes them essential in today's digital landscape, where attackers constantly evolve their strategies to avoid detection.

The principle behind neural networks, as proposed by Madhavaram et al. (2023) is that they consist of interconnected artificial neurons organized into layers that process input data using weights and activation functions. As data moves through the network, each layer captures increasingly abstract and meaningful representations. The lower layers handle basic features like word usage, while deeper layers detect more complex elements such as context and meaning. When applied to phishing detection, this process enables the network to identify subtle details in text, URLs,

metadata, or even the writing style of fraudulent emails, which are key indicators of a phishing attack. These networks are trained using large amounts of labeled data, such as phishing and legitimate emails or web pages, allowing the model to adjust its internal parameters to minimize prediction errors. With repeated exposure, the model becomes highly accurate at recognizing patterns in email subjects, headers, hyperlinks, and HTML that are often associated with phishing activities, thereby effectively distinguishing between safe and malicious messages (Safi & Singh, 2023).

Madhavaram et al. (2023) emphasized the value of neural networks for understanding phishing emails by identifying hidden connections among text elements, embedded links, sender metadata, and structural features that are not immediately apparent to human analysts. This was made possible by the ability of deep learning models to automatically learn and optimize feature representations. Traditional systems rely on manually extracted features by experts, such as URL length or domain age. However, neural networks eliminate the need for such manual effort because they automatically identify and learn these features. This represents a major shift in how cybersecurity analytics is conducted.

Safi & Singh (2023) built upon this approach by using neural networks to detect phishing URLs. Their study utilized GitHub data and achieved an impressive accuracy of 96.60%. Compared to the conventional Whois command, which is often slow and incomplete, deep neural networks can quickly identify relationships between URL segments, subdomains, and word formations with high predictive accuracy. This efficiency is especially crucial for real-time phishing prevention systems in browsers and email clients, where a few milliseconds can determine whether a user falls victim to an attack.

Zabihi-Mayvan & Doran (2019) made a significant contribution by developing a self-organizing neural network designed to detect phishing websites. Their model demonstrated the adaptability of neural networks, as they can restructure themselves in real-time to improve learning outcomes. The researchers tested the model using 17 features derived from 600 legitimate and 800 phishing websites obtained from Phish-Tank and Miller Smiles archives. Most of these features were based on external indicators such as domain validity, SSL certificate status, and content-based attributes.

The self-structuring network not only performed exceptionally well but also exhibited strong generalization capabilities, accurately classifying phishing sites it had never seen before or that belonged to different domains. This generalization is a key strength of deep learning systems because phishing attacks often involve new and altered elements, making them difficult for traditional rule-based or classical machine learning systems to detect. Neural networks, by contrast, learn and remember the underlying structure of data, which remains consistent even when surface-level information changes, making them highly resistant to adversarial attacks (Atawneh & Aljehani, 2023),

The other study that deserves attention was conducted by Atawneh and Aljehani (2023), who developed a system for detecting phishing websites using a feed-forward neural network, specifically a multilayer perceptron (MLP). They selected ten carefully chosen attributes as part of their dataset, which included factors like the age of a webpage, the presence or absence of certain words or symbols, and other metadata that reflected website behavior and design. The preprocessing stage involved dividing the dataset into training and testing sets, and converting

categorical features into numerical form so they could be used as input for the neural network.

Their MLP model achieved 93% accuracy on the test data and 95% on the training data. These results indicate a balance between model performance and its ability to generalize, suggesting that the neural network didn't overfit the data. The authors concluded that machine learning methods, like MLPs, can be effective for detecting phishing websites, outperforming traditional algorithms such as decision trees or logistic regression. They also noted that more advanced architectures, like convolutional neural networks (CNNs), recurrent neural networks (RNNs), or hybrid models, could further improve the system's performance (Atawneh & Aljehani, 2023),

When evaluating these findings, it's clear that deep learning has significant potential in phishing detection because it can handle both structured and unstructured data. Indicators of phishing can appear across different channels, including text content, HTML design, URLs, embedded media, and user behavior. Neural networks, especially deep architectures, are well-suited for processing these varied inputs in a parallel and hierarchical way. For example, CNNs excel at identifying local patterns in data, like n-gram combinations in text or pixel-level features in images. RNNs and their variants, like long short-term memory (LSTM) networks, are useful for dealing with sequential data, such as text or time-based URL patterns. MLPs, on the other hand, are effective for tabular or engineered data where relationships are complex but not necessarily sequential. This breadth of models offers researchers a versatile toolkit to tackle different aspects of phishing detection (Atawneh & Aljehani, 2023).

Theoretically, using neural networks for phishing detection marks a shift from rule-based systems

to data-driven models. Traditional filters were based on predefined rules, such as blocked domains or keyword-based heuristics. These systems were less flexible and required frequent manual updates. Neural networks, by contrast, can automatically learn and adapt to new threats. This adaptability is crucial since phishing attacks are constantly evolving, with attackers often changing URLs, mimicking trusted brands, or employing obfuscation techniques. Deep learning models can detect such patterns and provide a measure of confidence in their predictions, supporting real-time decision-making in email or website filtering. Recent improvements in transfer learning and pre-trained models like BERT, GPT, and their variants have further enhanced the ability of neural networks to understand linguistic intent and context, helping to distinguish between phishing and legitimate messages (Atawneh & Aljehani ,2023),

The performance results from studies such as those by Safi & Singh (2023) and Atawneh & Aljehani (2023) are particularly significant for demonstrating the accuracy and reliability of neural networks. Achieving over 90% accuracy on benchmark data shows these models can effectively reduce false positives and false negatives. A low false-positive rate helps ensure that genuine emails are not incorrectly flagged, maintaining user trust and productivity. Conversely, a low false-negative rate ensures that actual phishing attempts are caught and blocked. These benefits are even more pronounced when models are trained with advanced techniques such as dropout regularization, batch normalization, and adaptive learning-rate schedulers, which help improve generalization despite increased model complexity.

However, despite their effectiveness, neural networks presented certain challenges. One major issue is their lack of interpretability. The decision-making process involves thousands or even

millions of parameters, making it difficult for human analysts to understand exactly how a prediction is reached. This so-called black-box nature can be a problem, especially in cybersecurity, where transparency and accountability are essential. As a result, researchers are exploring explainable AI (XAI) methods in phishing detection, such as visualization tools, attention mechanisms, and feature attribution methods, to highlight which aspects of an email or website are most significant in a classification. These approaches enhance user trust and regulatory compliance, which is especially important in industries like banking and government communications (Atawneh & Aljehani, 2023),

The other issue is the quality and amount of data available. Neural networks that work well usually need a lot of different data to learn from. However, publicly available phishing data is often small, old, or only covers certain types of attacks. For example, the GitHub dataset used by Safi & Singh (2023) is well-organized, but it might not keep up with the fast-changing tricks used by phishing attackers. To solve this, researchers have suggested using data expansion, creating fake data, and methods that let models keep learning new information. They also suggest using systems that mix deep learning with rule-based or logic-based methods to get the best of both worlds automated pattern recognition and human-like understanding.

Zabihi-Mayvan & Doran (2019) proposed a self-organizing neural network, which is a good example of new model designs that help break away from traditional models. These models can avoid learning too much from a specific set of data by changing their structure based on the data's complexity, which makes them more efficient. This kind of self-adjusting behavior fits with bigger trends like neuro-evolution and AutoML, where models can evolve their structure and settings on

their own, without needing human input. This is especially useful for phishing detection because phishing websites change quickly as attackers try new ways to trick people. The ability for a model to change with new information helps keep the system stable over time and cuts down on the cost of keeping it updated.

Atawneh & Aljehani (2023) point out that feed-forward models like Multi-layer perceptron (MLPs) are still useful when the input data is carefully designed and not too complex. MLPs are effective and easy to understand, making them a good choice for organizations with limited computing power or small datasets. The authors focus on steps like converting categorical data into numbers and normalizing the data, showing how important data preparation is for making models work well. Their results show that neural networks can learn complex relationships better than simpler models like logistic regression, even with few input features.

Besides the results, all these studies highlight how important deep learning is in building systems to fight cybercrime. Neural networks are not just tools for classification, they are smart systems that learn and make decisions on their own. As phishing attacks get more complicated, neural networks are needed to spot zero-day attacks, which are new attacks that have not been seen before. There is also potential to use neural networks in bigger security systems like SIEM or intrusion detection systems to track threats in real time and understand the bigger picture. Neural models can also handle data from different sources, which helps detect cross-channel phishing attacks that happen across email, SMS, and social media (Sunyata-Madhavaram et al., 2023),

Overall, the work from Sunyata-Madhavaram et al. (2023), Safi & Singh (2023), Zabihi-Mayvan

& Doran (2019), and Atawneh & Aljehani (2023) shows a promising future for phishing detection using neural networks. Their flexible designs, ability to detect features on their own, and wide range of uses make them much better at handling the changing nature of phishing attacks than older machine learning methods. Although challenges such as data diversity, explainability, and costs remain, advances in explainable AI, transfer learning, and self-organizing models are likely to overcome them. In the future, we might see phishing detection systems that are fully intelligent, constantly learning new threats and keeping up with the fast pace of cybercrime. Deep learning, therefore, is key to the future of cybersecurity, offering both the theory and practical tools to protect digital environments from the ongoing threat of phishing.

vi. Fuzzy Decision Tree and Naïve Bayes

The email communication form of digital interaction has gained popularity among the most common types of digital interaction. It has consequently resulted in growth in spam and phishing attacks that are threatening to users and organizations. In this regard, researchers have created state-of-the-art machine learning and hybrid techniques to enhance the accuracy of spam detection and enhance the security of the email system (Ahmed et al., 2022).

The researcher Ahmed et al. (2022) suggested that another approach to spam detection is to use a combination of more than two algorithms to enhance the classification. This method incorporated a baking voting algorithm to detect patterns in spam behavior, particularly in the real world, where much of the characteristics of spam can hardly be observed. The ensemble techniques, like voting algorithms, synthesize the performance of multiple classifiers in such a way that one algorithm does not make the final prediction when using the information of just one algorithm. This line of

action enhances reliability since every classifier is involved in measuring trends in the data. Fuzzy Mamdani rules were also used in their system to include decision trees in order to distinguish between legitimate emails (ham) and spam emails.

According to Ahmed et al. (2022) a decision tree is a common classification method as it divides data into smaller branches as per decision rules by examining the features of data. Nevertheless, binary choices might not be the only way in a complicated email setup. Because of this reason, the fuzzy Mamdani rules were employed to enable fuzzy logic in the decision-making procedure to enable the system to deal with uncertainty and imprecision in email information. Fuzzy logic enables the classifier to put degrees of membership in classes, and thus enhances the flexibility and accuracy of the spam detection.

The Naïve Bayes classifier was used after the decision tree, using fuzzy logic rules, to further analyze the dataset. Naive Bayes is an efficient machine learning method based on Bayes' theorem that is widely used in spam filtering because it provides high accuracy for text classification. The algorithm uses a set of words, phrases, or patterns within the contents of the emails to determine the likelihood that the email is spam or legitimate. Although independence of features is assumed in Naive Bayes, which is not necessarily true in real-world settings, it still works well in most email classification problems. Naive Bayes was used in the suggested hybrid model to supplement the decision tree model by capturing probabilistic relationships among potentially overlooked features through rule-based approaches. After this, the voting outcomes were divided into smaller units, and the baking procedure was used to determine the optimal weights for the predictions of each classifier (Zuraiq & Alkasassbeh, 2019).

This weighting process enabled the model to reduce the weights of classifiers that had performed better during training. This way, the system was able to classify emails with greater accuracy. The study data comprised 1000 email samples: 650 (65) legitimate and 350 (35) spam. This distribution reflects the common imbalance in real-world email datasets, where valid email data usually dominates spam. Although the imbalance exists, the hybrid model performed better on classification, indicating that multiple algorithms may be used to achieve a high spam detection rate. In addition to this hybrid model, Ahmed et al. (2022) introduced another email categorization method supervised by a machine learning tool, tailored to Internet of Things (IoT) systems. Together with spam and phishing attacks, IoT devices are also vulnerable to internet-based attacks as more and more gadgets become increasingly dependent on the internet. The researchers thus proposed a Multiview learning strategy that aims to gather finer-grained information for the classification of emails. Multiview learning is a machine learning method that examines multiple collections of features derived from different views of the same dataset.

By analyzing information across varying perceptions, the model will be able to capture more patterns and relationships that would not be evident when a single set of features is considered. In this experiment, the Multiview strategy was used with both internal and external feature sets to create a two-view dataset. Internal features included information that was literally copied and pasted into the email, such as the subject line, message body, sender information, and email metadata. Such characteristics assist the system in scanning the material and format of the email. External features, on the other hand, are contextual information acquired from external sources such as domain reputation databases, blacklists, and network behavior indicators (Zuraiq &

Alkasassbeh, 2019).

The Multiview system was able to combine both internal and external features to give a more comprehensive dataset to classify. This strategy enhanced the spam detection capabilities of the system that tries to hide in the guise of a legitimate email. The suggested approach was experimented on two datasets in a real network setting, and the researchers were able to test the performance of the model in real-life settings. Real-network testing is necessary as email systems in the real world need to operate with large amounts of messages, which could be of different formats and languages, with different attack methods. The study results also revealed that the Multiview model was more accurate in classification than the conventional email classification models. Moreover, the researchers have compared the Multiview approach to some of the available models and established that it has always been better than them in the accuracy of detection and reliability. The other valuable addition to the area of research about phishing and spam detection is the Neuro-Fuzzy scheme that Zuraiq & Alkasassbeh (2019) proposed.

This method is a hybridization of fuzzy logic and artificial neural networks that form an intelligent system that has the ability to reason as well as learn through data. Fuzzy logic has a high level of success in dealing with uncertainty and linguistic variables that are typical in emailing. Nonetheless, the existing traditional fuzzy systems are based on a set of rules that are difficult to learn and do not operate with large datasets. Neural networks, on the other hand, can learn more intricate patterns using data by training them though they are usually not interpretable or able to reason in language (Zuraiq & Alkasassbeh, 2019).

The Neuro-Fuzzy system combines the power of fuzzy logic and neural networks; thus, it is said to advantage both methods. The neural network element gives the capability to learn patterns and relations in the data set. In contrast, the fuzzy logic element gives the system the capability to create linguistic principles and derive uncertain data. The researchers obtained 288 features in total from five large sources of input, which comprised legitimate site regulations, user behavior profiles, the PhishTank database, user-specific sites, and pop-up messages in email. All these inputs are valuable for phishing detection. A good example is legitimate site regulations that will help the system determine whether an email is attempting to impersonate trusted sites, such as banks or online services (Ahmed et al., 2022).

User behavior profiles can provide clues about how users typically interact with emails and websites, enabling the system to identify abnormal behavior that may signal a phishing attack. The PhishTank database is a repository of trusted phishing websites and URLs reported and confirmed as phishing by security experts. With this database incorporated into the detection system, the model can find known phishing links in emails in a short time. User-specific sites are those that the user often visits, e.g., social media or financial services portals (Ahmed et al., 2022).

Phishing email messages often mimic these websites to deceive users into providing sensitive information. Lastly, email pop-up messages are examined as a number of phishing attacks are based on the use of deceptive pop-ups that are used to acquire user login credentials or other sensitive information. The fact that the Neuro-Fuzzy system managed to extract these 288 features allowed it to consider both numerical and linguistic attributes of phishing attacks. Quantitative characteristics refer to quantifiable features such as the number of suspicious links, the domain

age, the length of URL, and the frequency of a particular keyword (Ahmed et al., 2022). Linguistic features Linguistic indicators occur in the form of text and requests that may be applied to personal information, grammatical structures that are usually connected with the phishing email.

The researchers also used a 2-fold cross-validation to determine the effectiveness of the planned system. Under this approach, the data is split into two equally sized subsets, one of which is used to train the model, and the other is employed to test it. The procedure is then repeated with the roles of the subsets reversed. Cross-validation assures that the model is evaluated on other groups of the data and lowers the chances of overfitting, and gives a more accurate representation of the performance of the model. The outcome of the experiment proved that the Neuro-Fuzzy system reached an accuracy rate of 98.5 percent in the case of analyzing the 288 features extracted. Such a high accuracy rate indicates that the use of fuzzy logic and neural networks is a good approach to identifying phishing attacks (Zuraiq & Alkasassbeh, 2019).

In general, the research by Ahmed et al. (2022) and Zuraiq & Alkasassbeh (2019) confirms the significance of hybrid intelligent systems in enhancing spam and phishing detection. By combining various machine learning algorithms, ensemble approaches, fuzzy logic, neural networks, and Multiview learning techniques, such systems can examine email data from multiple perspectives and identify more malicious patterns. With the ongoing development of cyber threats and the attackers becoming more advanced in their methods of circumventing the conventional security filters, the hybrid method will probably become progressively significant to the upcoming email protection systems. The methods are more flexible, more accurate, and more adaptable, and thus are appropriate to guard the contemporary digital communication landscape against spam and

phishing threats.

vii. Natural Language Processing

Phishing emails are one of the most common cybersecurity threats to individuals, organizations, and digital infrastructures around the world. Hackers are always coming up with advanced ways of deceiving users to disclose personal and sensitive information like passwords, bank accounts, and personal identifying details. Consequently, phishing email detection has become a significant research field in cybersecurity, data science, and artificial intelligence. Researchers have discussed many methods, such as machine learning, natural language processing, and deep learning, to enhance phishing attack detection and prevention (Salloum et al., 2022).

The research by Salloum et al. (2022) makes a valuable contribution to the research area, as it features a systematic review of the literature covering phishing email detection methods in detail. To study the development of research in the field of phishing detection, determine the main methodologies, assess the datasets applied in experiments, and emphasize the importance of natural language processing (NLP) and supervised machine learning (ML) algorithms in solving phishing threats, the researchers considered 100 research articles published from 2006 to 2022. Their work gives a clear view of the current research map, as well as identifies gaps that need to be filled by research, such as the incorporation of deep learning models, including convolutional neural networks (CNN) and long short-term memory (LSTM) networks, to enhance the detection of phishing emails.

The systematic review by Salloum et al. (2022) examined key areas of research on phishing email

detection. Such includes the properties of phishing email identification, the datasets and resources utilized in experimental research, the evaluation measures that are applied to determine the model performance, and the machine learning and natural language processing algorithms used in the phishing detection systems. By reviewing studies published over more than 15 years, the authors traced the evolution of phishing detection techniques and assessed the impact of advancements in artificial intelligence on the domain. Among the most important findings of the review, there is the fact that the research on phishing detection has been gradually moving away from the traditional rule-based systems to the more advanced machine learning-based methods and NLP-based ones that have the capability of processing complex email content and can discern subtle patterns that are related to phishing attacks.

Among the greatest factors of phishing email detection lies the ability to identify features of distinction that can distinguish phishing emails from legitimate emails. As Salloum et al. (2022) state, phishing email characteristics are typically divided into a few categories, namely, textual, structural, behavioral, and network-based features. Textual features are features that are contained in the body of the email, which include the subject line, body text, embedded hyperlinks, and linguistic patterns. Phishing emails are usually done in a convincing language to either make the receiver feel urgent or fearful.

As a case in point, attackers can utilize such phrases as your account will be suspended, verify your account immediately, or an urgent security update is necessary to force users to take instant action without investigating the validity of the message. Such linguistic indicators present meaningful hints that machine learning models can be developed to identify phishing efforts. The

analysis of such textual features is especially useful with natural language processing techniques since they provide a system with the opportunity to process and interpret the semantic and syntactic structure of email messages (Salloum et al., 2022).

Structural features, in their turn, are the technical characteristics of the email message, i.e., the header information, the sender's address of a sender, the domain name, and the email format. Inconsistency in such aspects is common in phishing emails. As an example, the attackers can try to spoof email addresses that are similar to trusted domains or construct URLs that appear as if the viewer is visiting reputable websites. This might include the use of similar-looking characters instead of letters, e.g., using the letter 1 instead of the letter l in a domain name. Machine learning algorithms can use these structural features to extract suspicious patterns that can be used to detect phishing activity. Moreover, network-related features like IP address reputation, domain registration details, and the location where the email is hosted can also be of help in identifying the malicious emails. These characteristics enable the system to confirm that the sending domain is correlated with previous phishing attacks or network malpractices (Salloum et al., 2022).

Another significant element that is taken into consideration in the research on phishing detection is behavioral features. These attributes are concerned with user and email system interaction, whereby the number of times a user clicks on links in emails, the nature of the websites which a user usually visits, as well as the overall communication habits, are used. Behavior-based phishing detection techniques can be used to detect abnormal behavior that is not reflective of the behavioral pattern of a user. A case in point is when a user suddenly tries to access a site he or she does not know about through a link included in an email message; the system would identify the activity as

suspicious. Machine learning models can provide a more detailed picture of a phishing attack by integrating behavioral analysis with textual and structural information (Salloum et al., 2022).

The second important aspect analyzed by Salloum et al. (2022) was the use of datasets and resources in phishing detection research. Datasets are important in both training and testing machine learning models, as they provide the examples the system needs to learn patterns related to phishing emails. The researchers observed that most research works are based on publicly available datasets of legitimate and phishing emails.

The Enron email dataset, the Nazario phishing corpus, and other maintained phishing email archives by security researchers are common datasets. Such datasets usually include a thousand email messages, classified as legitimate or phishing, allowing a supervised machine learning algorithm to learn classification patterns. Nevertheless, the difficulties with dataset availability and quality were also mentioned in the review. Among the core challenges is that phishing attacks evolve rapidly. Thus, previous datasets might not be a good representation of the latest phishing strategies employed by the attackers. Consequently, the performance of models trained on outdated datasets may be poor when applied to real email systems. Also, most of the datasets are unbalanced, with legitimate emails being significantly more than the phishing messages. Such an imbalance may result in biased classification models that can hardly differentiate minority classes like phishing emails (Salloum et al., 2022).

Another important point that is evaluated in the systematic review is the evaluation of phishing detection systems. The commonly used performance measures to gauge the effectiveness of

models by researchers are accuracy, precision, recall, and F1-score. Accuracy is the count of valid classifications made on emails withdrawn in contrast to the count of the total emails in the set. Although accuracy is a handy measure, it does not necessarily give a full view of the model's performance, particularly when the datasets are imbalanced. Indicatively, when a dataset has many valid emails and a few phishing emails, a model that merely classifies all the emails as valid may still have a high accuracy, but will not successfully identify the phishing emails. In order to overcome this shortcoming, precision and recall are used as complementary measures by researchers. Precision is defined as the rate of correct phishing emails that have been labeled as phishing (where all were phishing). In contrast, the recall is defined as the rate of phishing emails that the model correctly labels as phishing (where all were phishing). The F1-score is a factor that integrates precision and recall into one parameter that gives a moderated evaluation of classification effectiveness.

A major part of the systematic review conducted by Salloum et al. (2022) was devoted to natural language processing and machine learning algorithms for detecting phishing emails. NLP methods allow computers to study and understand human languages; therefore, they are well-suited to analyzing email messages. Some of the common NLP approaches applied in phishing detection are tokenization, leaf removal, stemming, and term frequency-inverse document frequency (TF-IDF). These methods convert textual information into numerical values that can undergo machine learning to be processed. After mining the textual characteristics, machine learning algorithms like Naive Bayes, Support Vector Machines, Decision Trees, Random Forests, and Logistic Regression can be applied to label emails as being phishing or legitimate. High detection accuracy with these conventional machine learning methods has been reported in many studies, especially when used

together with carefully developed feature extraction methods.

Although the conventional machine learning algorithms have achieved success, Salloum et al. (2022) highlighted that a gap exists in the literature that lacks systematic literature reviews, in particular on natural language processing methods in phishing email detection. The researchers claim that the current reviews tend to focus on more general issues of cybersecurity or phishing detection in general, without analyzing the NLP-based methods in detail. This gap shows the necessity of conducting additional studies that could specifically be focused on the use of advanced language processing methods to enhance phishing detection tools. This kind of study might examine the application of superior NLP algorithms that will be able to discern context, semantics and linguistic intricacies in email messages.

The other significant contribution of the review is the demonstrated value of deep learning techniques in detecting phishing emails. Deep learning is a branch of machine learning that uses neural networks with multiple layers to identify complex patterns in data. Deep learning models have the ability to learn the features of interest naturally without the need to do feature engineering, as is the case with conventional machine learning algorithms. This feature qualifies them especially to process textual data that is highly voluminous such as in email messages. Salloum et al. (2022) have pointed out the perspective of deep learning models being used in phishing detection, including convolutional neural networks (CNN) and long short-term memory (LSTM) networks.

Convolutional neural networks have been extensively applied in processing images but have been

effectively applied in text classification issues. When applied to phishing detection, CNN models can encode the word sequence in email messages to recognize patterns associated with phishing attacks. Convolutional layers in the network learn local features of the text, and pooling layers reduce dimensionality while retaining the most significant information. CNN models can detect increasingly complex patterns in email text by stacking convolutional layers upon convolutional layers. This capability will enable the model to identify nuanced linguistic symptoms that may indicate phishing (Akinyelu & Adewumi, 2023; Hussain et al., 2023; Alhaj et al., 2023).

Instead, long short-term memory networks are a type of recurrent neural network used to process sequence data. LSTM networks perform especially well at capturing long-term dependencies in text sequences (Alzahrani & Alenazi, 2021). To identify phishing communication patterns in an email message, LSTM models can be applied to analyze the sequence and context of words in the message to determine whether it has been used to communicate phishing. As an example, an LSTM model can identify trends in how phishing emails format their messages, e.g., the combination of urgent-sounding language and a demand to provide personal information. LSTM networks can produce more precise classification outputs than models that process individual words and ignore the context of whole sentences and paragraphs.

Deep learning methods can also be integrated into phishing detection systems, enabling the use of word embeddings represented as numerical vectors capturing word-level semantic relationships. The common parameters of word embedding, including Word2Vec, GloVe, and fastText enables the models to comprehend how words and phrases are similar. As an illustration, the model can be trained that the words verify, confirm and authenticate frequently occur in phishing email

messages, which seek user authentication. These semantic relationships can be used to enhance deep learning models, which can, in turn, recognize phishing messages, even when the attacker tries to mask their message in slightly different words (Alzahrani & Alenazi, 2021).

Although deep learning models have potential, Salloum et al. (2022) indicated that their use in phishing email detection studies was comparatively limited compared with conventional machine learning techniques. The first reason could be that training deep learning models is very computationally expensive, as they typically require large datasets and strong hardware. Also, more sophisticated machine learning models are potentially harder to interpret than simpler ones, and thus, security analysts might struggle to learn how classification decisions are made using deep learning. However, the researchers have stressed that there are promising applications of deep learning: improved accuracy and the ability to extract features automatically make this field an essential focus for the future.

The review finally concluded that there is need to conduct more research to investigate use of advanced deep learning methods in detecting phishing emails. More specifically, the researchers recommended that the future research should be directed at creating CNN-based and LSTM-based models that would be able to analyze large volumes of email data. It is possible that such models can be combined with NLP methods to make more advanced phishing detection systems that can comprehend the language content and structural characteristics of email messages. Furthermore, it could be possible to enhance the performance of deep learning models through the use of other methods, including ensemble learning, and behavioral analysis to enhance detection performance (Thakur et al. 2023).

In identifying phishing, Thakur (2023) proposed a multi-spatial CNN system. They launched a real-time prototype of Raspberry Pi. Their solution got an actual detection rate of 86.63 percent and a 30 percent reduction in the time taken to run the process. Their study simply considered the use of a single deep learning (DL) model without comparing its performance to other State-of-the-art (SOTA) methods. Their study suggests the need for further research into a DL-based webpage content phishing detector for resource-constrained sensors (Thakur et al., 2023).

Thakur et al. (2023) also proposed a framework for spam and phishing detection using DL and conventional ML algorithms and conducted experiments on email and URL datasets, training for 1000 epochs with different learning rates. They asserted that their suggested framework could uncover malicious activities, and that its functionality could be enhanced by adding sub-modules for DNS malware and log analysis, which they considered an important direction for the future.

The study by Thakur et al. (2023) proposed a multidimensional feature-based method for phishing detection using DL. Their approach implied extracting character sequence features and performing fast classification. Their proposed procedures demonstrated high accuracy of 98.99 and low FPR of 0.59. Their research had a limitation in that it only discussed one form of phishing attack (website-based) and never touched on the other forms of phishing attacks which could utilize alternative methods and approaches. To enhance the technique, the authors suggested to use other deep learning techniques to extract text and code features in web pages in further research.

viii. Long Short-Term Memory and Artificial Neural Networks

The article by Adebowale et al. (2023) provides substantial information on the dynamic nature of the relationship between conventional machine learning tools and new deep learning algorithms in phishing detection, particularly when systems rely on URL-based approaches for prediction. Their study specifically tested the Long Short-Term Memory (LSTM) network as an abstract component of an overall model aimed at identifying phishing websites. The study was fueled by the growing sophistication of phishing attacks, in which malicious individuals continuously change website names and domain hierarchies to evade traditional rule-based or fixed machine learning models.

Adebowale et al. (2023) aimed to establish how sophisticated neural networks, such as LSTMs, can outperform less sophisticated algorithms, such as Random Forests (RFs) that rely heavily on handcrafted features. The authors, in their research, compared an RNN-based model (with an LSTM core) with a Random Forest classifier, using a shared set of 14 lexical and statistical URL features. These attributes were carefully chosen to observe the underlying trends that can distinguish a legitimate site and a phishing one, thus providing a moderate measure between the traditional and deep learning paradigms.

Parameters used in their study in lexical and statistical aspects were: length of URL, number of subdomains, special characters in the URL, e.g., @, -, and underscore, ratio between digits and letters in the domain name, use of HTTPS, domain age, and entropy, among others. All these features are famous signs of the phishing motive. As an example, phishing URLs can be characterized by a tendency to be longer in length, with an abnormal number of subdomains, and

with the use of deceptive brand names to trick internet users into thinking it is a trustworthy one. Similarly, characteristic elements, like the use of HTTPS and age of domain, are vital since phishing sites are typically temporary and can be uncertified with regard to the use of the Secure Socket layer. Through the analysis of these 14 features, Adebowale et al. (2023) hoped that they would be able to create a complete report of the structural and lexical composition of URLs. In this way, they could easily contrast the performance of manual feature engineering (as in Random Forests) with the automatic feature extraction features of deep learning models such as LSTM.

Random Forest (RF) models were used as the machine learning baseline in their experiments. Random Forests are an ensemble learning algorithm that builds many decision trees during training and returns the mode of the class labels (in classification tasks). They are characterized by their strength and ability to accommodate non-linear relationships among features, and are a popular option for phishing detection using manually engineered features (Thakur et al., 2023).

The 14 handcrafted features used in the RF model by Adebowale et al. (2023) were the input features, which are interpreted easily and readily computed. Nevertheless, RF is not without its drawbacks. In larger-scale or feature-constrained systems, RF has an inherent weakness in that it cannot dynamically learn new representations based on new data without first re- engineering its features. RF models are constantly brought up to date, or their feature set re-defined, as phishing strategies may develop (with obfuscation, or homograph attacks, or URL shortening). This weakness highlights the main rationale of studying LSTM-based architectures in detecting phishing, which can learn to represent data autonomously with time (Thakur et al., 2023).

Adebowale et al. (2023) used the Long Short-Term Memory (LSTM) model where the input was the URL strings in the form of sequences and the model used the contextual dependencies among characters and tokens. There is also a difference between LSTMs and Random Forests in that the latter relies on features that are fixed and static, whereas the former uses sequences of characters per URL and learns temporal relationships between them.

This methodology enables the model to retain patterns including repeated brand-name insertions, misleading word combinations or manipulation at domain levels that change with time. Indicatively, phishing websites usually replicate trusted brand names (e.g., www.paypal.verify-login.com) -a.characteristic LSTM model is able to learn through positional dependencies and character distributions that are out of place when compared to real URLs. The gating process in LSTM, which consists of an input, forget and output gates, assists in memorizing essential sequence data and forgetting unimportant data; which avoids vanishing gradient problems inherent to standard recurrent neural networks (RNNs). This renders LSTMs very appropriate in sequence based phishing detection whereby subtle contextual information is used to tell whether a URL is malicious or not (Thakur et al. 2023).

The relative performances illustrated by Adebowale et al. (2023) showed that LSTM-based models were much better than the Random Forest classifier, as they performed at higher accuracy, precision, and recalls in a variety of testing data sets. The authors attributed this to the fact that LSTM was able to automatically learn the complex nonlinear dependencies of URL sequences by extracting hierarchical representations of URL sequences without the need to craft features manually. Random Forest models, on the other hand, although interpretable and relatively quick

to train, were limited by the fact that they needed a fixed set of manually determined features, and as such, they could not extend their capability to new phishing patterns that are emerging. Furthermore, the Random Forest method was not able to handle adversarial URLs: the URLs that are being generated on purpose and that are meant to appear as valid syntax, but which contain malicious variants. However, the LSTM could identify these patterns because of its sequential learning abilities, which identify both the micro-level (character-based) and macro-level (word-based) organizations (Thakur et al., 2023).

A key conclusion of the study was that feature engineering is less important than in scenarios where deep learning models are utilized, as models such as LSTM can learn feature representations from raw data. The identified observation represents a significant paradigm shift in the research on phishing detection, i.e., the shift toward automated learning of features, as opposed to manual feature selection. In classical algorithms like RF, Naive Bayes, or Logistic Regression, the performance depends heavily on the selection of features. Researchers need to determine the most predictive characteristics of URLs and derive numerical or categorical values from them. This kind of process, however effective in structured problems, creates bias and hinders the expandability of the detection systems (Thakur et al., 2023).

Deep learning structures such as LSTM alleviate this problem by taking raw data (here, character sequences) directly and finding learned discriminative features by learning through their layered structure. Such an ability to learn high-level abstractions enables LSTM models to be useful in changing phishing strategies without necessarily requiring explicit redesigning of features, hence they can be more robust to changing threat environments (Thakur et al., 2023).

The research also highlighted the significance of the quality and diversity of data in the analysis of the model performance. The data set, which was applied by Adebowale et al. (2023), included thousands of URLs that were selected by a variety of sources, such as valid domains and verified phishing websites. The authors balanced the classes in order to avoid training with a biased dataset because an unbalanced dataset (when the set of legitimate emails is much larger than the set of phished emails) can result in a misleadingly high accuracy and low performance in the real world. The recall rate of the minority phishing example was high, which was due to the fact that LSTM was robust in modeling uncommon but significant sequences. However, the random Forest models had greater variance when subjected to new, never-seen URL structures, which is an indication that they are reliant on pre-established characteristics that are unlikely to detect new phishing patterns. These findings demonstrate that deep learning not only increases the accuracy but also generalization and adaptability, which are essential in cybersecurity applications where the threat patterns keep changing.

The other piece of interesting information introduced by Adebowale et al. (2023) was their discussion of computational complexity and resource considerations. Although LSTM models were more accurate, they required much higher computation and inference to train and predict compared to the Random Forests. LSTM Layer Training: LSTM models can be trained on sequential data with significant matrix operations and gradient propagation across time steps, which consume more time and memory.

Nonetheless, Adebowale et al. (2023) mentioned that this computer price is compensated for by

the predictive ability gains and flexibility. In real-world deployment, transfer learning or less complex models can be used to optimize such models, to achieve a tradeoff between efficiency and accuracy. Random Forests, on the other hand, are also not as attractive in resource-constrained settings, e.g., in small-scale organizations or embedded systems, because they have a low computational footprint. Thus, Adebowale et al. (2023) concluded that the choice of models must be based on the operational setting, i. e., deep learning when working with large-scale adaptive detection systems, and traditional machine learning when the environment is constrained or interpretability oriented

The research also provided some useful information on the topic of interpretability versus performance trade-offs of phishing detection. Although LSTMs were particularly accurate, they are black-box models, which makes it hard to understand why a particular URL was classified as phishing or legitimate. Random Forest models, on the other hand, are able to give clear decision directions based on feature thresholds, allowing them to be explained and debugged more easily. Such interpretability can be a necessity in the regulatory or enterprise environment, where transparency is mandatory, e.g., adherence to data protection or cybersecurity reporting regulations (Adebowale et al., 2023).

Adebowale et al. (2023) suggested that next-generation systems could integrate the advantages of the two methods: initial detection and feature discovery with LSTM models and interpretable decision-making and auditability with Random Forest models. This mixed research design is in line with larger tendencies in cybersecurity studies, in which mixed systems integrate several algorithms to deliver accuracy and transparency.

Respecting the methodological aspect, the presence of hybrid measures of evaluation in the study by Adebowale et al. (2023) contributed to the validity of the results. The authors compared accuracy only with precision, recall, F1-score, and area under the ROC curve (AUC) instead of focusing solely on accuracy, which gives a comprehensive view of the model performance. The LSTM model scored higher in values of recall, i.e., it was more effective in identifying phishing URLs without missing the potential threats. This plays a critical role in cybersecurity, where a false negative (phishing missed) can be disastrous to security, such as loss of data and money. In a few instances, the precision shown by the random Forest models was a little higher, meaning that there were fewer false positives, but the overall recall was lower. This tradeoff between accuracy and recall highlights one of the most frequent problems in phishing detection, which is to strike a balance between sensitivity and realistic feasibility. Systems that are too sensitive may produce profuse false alarms, and those that are not sensitive enough may still allow phishing emails to pass.

Adebowale et al. (2023) also emphasized that the generalizability of such deep learning models as LSTM is due to their capability to train latent representations, which can be transferred to other datasets and fields. An LSTM model can be trained and then fine-tuned with only a small amount of extra training to follow new phishing patterns once it has been trained on new data. This feature provides a considerable benefit over the Random Forests, which have to be trained afresh whenever the distribution of the features varies. The flexibility of the LSTM therefore renders it suitable for real-time phishing detection systems, which continuously learn or update through online learning. Another finding by the researchers was that the character-level embeddings used

in their LSTM implementation were useful to capture very fine-scale differences, including minor typographical differences in brand names (e.g., facebook.com or paypa1-login.net), which traditional models do not tend to notice. These results support the fact that the sequence-based deep learning models have a better representational strength on complex text-like information like URLs.

Within the larger scope of phishing detection studies, the results of Adebowale et al. (2023) are consistent with the conclusions of previous studies conducted by Kyaw (2024) ,as both researchers highlighted the benefits of deep learning models over traditional machine learning methods. The unified evidence of theirs is that the future of phishing detection is in the architectures that can auto-train the features, understand the context, and adapt to changes. These developments, however, also create new problems, including the requirement of large datasets (in order to be high-resolution), the risk of overfitting to certain data distributions, and the need to have interpretable AI mechanisms able to explain model decisions in sensitive cybersecurity usage. In that regard, although LSTM and other models have proven to be better in terms of detection capabilities, researchers and practitioners have to strike a balance between accuracy, transparency, and efficiency when applying them to operational phishing defense systems.

Summing up, the article by Adebowale et al. (2023) presents strong evidence to support the conclusion that deep learning technology, especially LSTM networks, outwits the conventional feature-engineered models, such as Random Forests, in detecting phishing websites, depending on the URL as the input. Using 14 lexical and statistical characteristics, the authors revealed the advantages and constraints of the classical machine learning, as well as revealing the

transformational potential of the sequence-based neural networks. Their study highlights a radical change in cybersecurity analytics: the manual and human-understandable models to autonomous and data-driven architectures, able to respond to the quickly evolving threat environment. In the end, their results confirm that LSTM and other deep learning models can not only improve their detection performance but also provide the prospect of more intelligent, self-evolving, and context-aware phishing detection models capable of ensuring their safety in an ever-more digital and dangerous world.

Sahingoz (2018) demonstrated in his research that most anti-phishing systems rely on blacklisting tactics, have high false-positive rates, and cannot detect new phishing attacks. They attempted to overcome this limitation by training the system with machine learning-based techniques, particularly Artificial Neural Networks (ANNs) and Deep Neural Networks (DNNs), and by looking at web page URLs to identify anomalous requests.

Whitelists, which are collections of legitimate emails, are used by list-based algorithms to identify fraudulent URLs. Malicious URLs are compiled into the blacklist. The system compares a user-input URL with a pre-existing list of URLs. Additionally, the user will be permitted access to the page if the entered URL is present in the list of valid URLs (Singh, 2020).

Visual correlations were used to identify fraudulent websites. This is achieved by using visual cues to divide the URL into several sections. Several levels, including block level, form level, and layout, are used to measure the (visual) resemblance (Singh, 2020).

As reported by Adebowale et al. (2023) the new algorithm that the researchers developed, called

the Intelligent Phishing Detection System (IPDS), is a hybridization of CNN and LSTM, through which the sites and their textual content are considered, which implies that the new algorithm comprises a greater number of elements to identify an activity as phishing. To the best of the researchers' knowledge, the most integrated text, image, and frame characteristics have not been studied using a deep learning-based (CNN-LSTM) phishing detection solution (Adebowale et al., 2023).

In several studies, character-level representations of URLs have been used as input features of deep learning models. All URLs are represented as sequences of characters, and the model is trained to identify patterns and characteristics based on these raw sequences. The merits of character-level representations include the fact that they can spot very small details and subtle patterns in the URLs, which can be critical for spotting complex phishing attacks (Kritika, 2024).

According to Yang et al. (2021), it uses the character embedding method, in which the URLs are embedded through expansion. Character embedding is better than word embedding due to the following reasons. The fixed number of characters does not allow for the absence of features due to the insertion of new words. URLS are not language dependent because one can apply them on any language because they employ the use of characters. The URL of phishing websites usually contains the phrases which are not rational. The data was pre-prepared in one-hot encoding before analyzing a number of machine learning (ML) and deep learning (DL) models with consideration of a hyper-parameter analysis. The optimum CNN model was achievable at the filter size = 7, the context = 100, the embedding window = 80, the pooling size = 4. It has a 95.97 accuracy of identification. It was found that dropout was done on both the concealed nodes to and without the

provision of the most effective accuracy over Long-Short Term Memory (LSTM). The accuracy encoding was more effective in CNN with one hot encoding when it came to dropouts; Word Embedding ruled out all other paradigms (Rathee & Mann, 2022).

Peng et al. (2019) introduced the CNN-LSTM attention-based model to identify and detect malicious URLs. Based on his experiments, this method detects malicious URLs with a 98.26% higher accuracy than shallow neural networks and single deep neural networks that were classified using global features and a Softmax classifier. In order to achieve strong linear separability of the features, this model relies on extracting and filtering the texture information of the URL to be identified, the statistical attribute information of the URL string, and the WHOIS (database) information, where these features are then encoded. To highlight important characteristics and optimize their pooling for global features, the CNN-LSTM model is then coupled with an attention mechanism (Peng et al., 2019).

Kritika (2024) was able to point out that, based on knowledge from secure networks, it is believed that distinctive feature designs for LSTM are necessary to address classification problems. Two feed-forward layers, three bidirectional layers, and a Softmax layer for predictions make up an LSTM's architecture (Kritika, 2024). This fully linked architecture enables the use of the inherent correlations between links. The connection undergoes co-occurrence exploration to learn from input attributes, and backpropagation to the LSTM layer is performed for more efficient learning (Kritika, 2024).

Attackers employ minor changes to the URLs to fool users on phishing websites, which usually

imitate the URLs of trustworthy websites. For example, the same letters are used to change google.com to gooIe.com, and "l" is changed to "I." Character-level embedding can more successfully detect minute changes in URLs, and adding additional information at the character level can improve the detection effectiveness of phishing sites (Yang, 2021).

ix. Convolutional neural networks and bidirectional Long Short-term memory

In summary, email phishing has seen a long-overdue improvement in recent times, thanks to the development of deep learning techniques. Deep learning models, particularly those that use both Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (LSTM) networks, have shown great potential in detecting phishing emails. CNNs are especially good at identifying local features in text, such as n-grams and common patterns found in phishing emails, like "click here" or "verify account" (Kyaw, 2024)

CNNs are also effective at analyzing different levels of email content, from characters to words, making them more capable of detecting phishing attempts. This was highlighted in Kyaw et al. (2021) who noted that CNNs excel in analyzing email headers and body content at various layers.

Maneriker et al. (2021) emphasized that CNNs can learn text embeddings from individual characters, which helps in identifying subtle patterns in phishing emails. This includes spotting unusual punctuation, spelling mistakes, or deliberate obfuscation techniques that attackers use to bypass traditional filters. The study also noted that such techniques often involve inserting random symbols, mixing uppercase and lowercase letters, or using characters from other alphabets, which are common in today's phishing attacks.

Kyaw et al. (2024) pointed out that while transformer models are better at understanding semantics and handling multiple contexts, they come with higher computational costs and face issues with token length. In contrast, CNNs can perform better on highly obfuscated or character-manipulated datasets. However, transformers excel when a nuanced understanding is crucial. To overcome these limitations, CNNs are often combined with LSTM networks, which help capture the semantic relationships and sequential dependencies in text. Zhang and Lee (2019) demonstrated the effectiveness of Text-CNN in detecting phishing indicators in email content, achieving high accuracy in identifying fraudulent language.

Bahnsen et al. (2017) highlighted the potential of sequence-based models like LSTM in phishing detection. Wang et al. (2021) took this a step further by developing a hybrid CNN-LSTM model. This model outperformed standalone models in accuracy and resilience, particularly in detecting sophisticated phishing techniques. The model used CNN for initial feature extraction and LSTM for sequence modeling, and it was tested on datasets such as PhishTank and SpamAssassin, showing strong performance with 97.8% accuracy in detecting both text-based and URL-based phishing threats.

Alom et al. (2020) used CNN for spam detection, combining text with metadata to achieve high accuracy in spam classification. However, the model's performance dropped when only textual data was used. Feng et al. (2018) improved CNN by using Word2vec embeddings on a Sina Weibo dataset, achieving 91.36% accuracy but facing challenges related to model complexity. Seth and Biswas (2017) combined CNN with both text and image data, achieving 98.11% accuracy, proving the model's adaptability across different input types.

Shahariar et al. (2019) proposed a hybrid CNN with LSTM and Word2vec, achieving 94.56% accuracy. Atawneh & Aljehani (2023) suggested that a deep learning model combining text analysis and machine learning could improve phishing detection. Their model, developed using a Graph Convolutional Network (GCN), was tested on a fraud dataset with both genuine and fraudulent emails, showing 98% accuracy and a very low false-positive rate of 0.015%.

Rhanoui et al. (2019) focused on developing an effective deep learning model for document-level processing and classification. They proposed a CNN-LSTM model based on Doc2vec for document-level sentiment analysis. The model was tested against other models like CNN, LSTM, LSTM, and CNN-LSTM, and it outperformed them, achieving 90.66% accuracy in classifying French press articles. CNN was best suited for analyzing or identifying patterns in suspicious word sequences in phishing, where it is trained on character-level or word-level embeddings while LSTM captures dependencies and context in email content which can be used to detect email intent based on context flow in sentences.

Thakur et al. (2023) have presented a DL-based phishing email classification system that was more effective than the rule-based and machine learning-based systems. It was demonstrated that the proposed architecture with CNN, MLP, and RNN, along with Word2vec embedding were highly accurate at the stage of training, but the accuracy of the test was rather low because the dataset was highly unbalanced (Thakur et al., 2023).

Their architecture was made up of CNN, MLP, and RNN with the inclusion of Word2vec embedding, which demonstrated high accuracy throughout the training; nevertheless, the test accuracy was relatively poor due to the highly skewed nature of the dataset (Thakur et al., 2023). They employed two data sets in their work. The first one was subtask 1, and the other was subtask 2. According to them, CNN was better at subtask 1 with an accuracy of 95. 2%, whereas RNN was better at subtask 2 with an accuracy of 93. 1%. Their proposed research paper emphasized the fact that the problem of the imbalanced datasets issue requires further study.

Thakur et al. (2023) introduced an algorithm that used a Neural Network reinforced with reinforcement learning to detect phishing attacks in real-time. Over the years, the proposed approach has been modified and enhanced through reinforcement learning and is able to deal with emerging phishing actions. Their suggested approach was refined and enhanced with the course of reinforcement learning and is capable of managing emerging phishing patterns (Thakur et al., 2023). They conducted a few tests on known data sets and reported a TPR of 99. They conducted several tests on well-known datasets and reported a TPR of 99. 07%, an accuracy of 98. 63%, a low FPR of 81%, an FNR of 93%, and a TNR of 98. 19%. They have concluded that in future studies, the offline dataset can be expanded to enhance the performance of the model and be applied to detect phishing, spam, and ham mail (Thakur et al., 2023).

Convolutional Neural Network Architecture

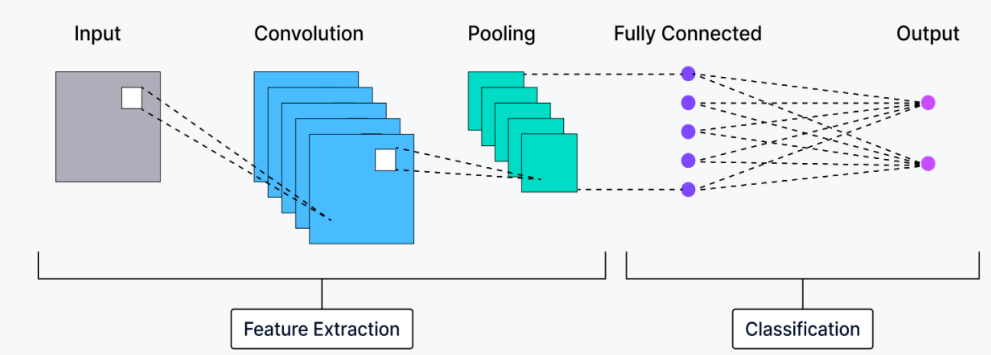

Figure 2-2 CNN Architecture

The CNN architecture: Feature extraction layer is in charge of extracting meaningful groups of data provided in the input layer; for example, URLs, Domain name, and so on. All the retrieved features are mapped to the appropriate group by use of a classification layer, e.g., as phished text/URLs or legitimate text /URLs. To make the data more compact, a pooling layer is frequently added to decrease complexity and speed up processing. The pooling layer helps in preventing overfitting by removing unnecessary details and noise, allowing the model to generalize better to new data. The output is then produced by feeding the final filtered data.

The studies that were reviewed show that CNNs are good at finding local and character-based patterns in phishing attacks, while LSTM and LSTM models are effective at understanding the sequence and context of these patterns (Kyaw et al., 2020, 2021, 2024; Maneriker et al., 2021; Bahnsen et al., 2017).

Wang et al. (2021) and Shahariar et al. (2019) also showed how combining CNN and LSTM models can improve phishing detection. This study builds on these methods by using a CNN-

LSTM model on URL and text data, bringing together local pattern recognition and bidirectional context understanding in one system for classifying phishing attempts.

x. LSTM with Self-Attention and Transformers

In summary, LSTM models with self-attention mechanisms have proven to be very effective in retrieving contextual information, especially for tasks like identifying spam. Xu, Zhou, and Liu (2021) used a self-attention LSTM along with ALBERT to work with Twitter and Weibo data and achieved a high accuracy of 91% and an F1-score of 90%. Similarly, Shahariar et al. (2019) demonstrated strong performance with a LSTM model that combined CNN and word2Vec, achieving an F1-score of 95.2% and accuracy of 95%.

Self-attention in LSTM networks and transformer embeddings have proven effective for detecting spam and phishing messages in modern communication systems. These models combine sequential learning, contextual representation, and attention mechanisms to capture complex language patterns. Researchers such as Xu, Zhou, and Liu (2021) and Shahariar et al. (2019) have shown that hybrid models using LSTM, CNN, and transformer embeddings achieve high classification accuracy and strong F1-scores. As cyber threats become more complex, deep learning methods are essential in developing smart and adaptive spam detection systems that protect users from phishing attacks. Transformer-based models also have a positive impact on cyberspace.

Uddin et al. (2026) evaluated RNN and BERT models to detect phishing emails, especially with uneven data distribution and growing data volumes. They found that BERT achieved a test

accuracy of 96.1%. Another study by Uddin et al. (2026) looked at various deep learning models, including CNNs, LSTMs, RNNs, and BERT, to classify emails as legitimate or phishing. BERT-based and LSTM-based models achieved an impressive accuracy of 99.61% across four public datasets, outperforming other solutions. Variants of BERT, like DistilBERT, RoBERTa, or Tiny BERT, are also gaining traction in phishing and spam research. Uddin et al. (2026) introduced the Improved Phishing and Spam Detection Model (IPSDM), which uses fine-tuned DistilBERT and RoBERTa, and offers high accuracy in their research. They also proposed Context-Aware Tiny Bert (CatBERT), which is faster and more resilient to adversarial attacks, especially those involving deliberate keyword substitutions. CatBERT achieved a detection rate of 87%, which is better than other models. Additionally, BERT-based models have been fine-tuned for detecting phishing URLs.

An et al. (2024) highlighted that the strong sequential data processing capability of Transformers allows Finsformer to uncover long-term dependencies and complex patterns in financial data.
The cluster-attention mechanism further improves the model's ability to extract critical patterns from financial transaction data, especially when working with large-scale datasets. Experiments show that Finsformer outperforms traditional deep learning models like RNN, LSTM, and BERT in several performance metrics. However, the model has its limits. One major issue is computational efficiency, especially with large datasets. Even though the Transformer structure supports parallel processing, training and inference speeds might slow down with big data. The quality and variety of data also greatly affect model performance. In real-world applications, financial data can be noisy, inconsistent, or incomplete, which impacts model accuracy and reliability.

Despite these strengths and limitations, there are further considerations for understanding Finsformer better and its potential use in financial data analysis. One key feature of the model is its generalizability across various financial situations. While Finsformer performs well on benchmark data, its effectiveness in other financial contexts, such as fraud detection, credit scoring, and stock market prediction, needs to be verified. Financial data is highly diverse in structure, volume, and dynamics, which can pose challenges even for advanced models like Finsformer (An et al., 2024),

According to An et al. (2024), transformer-based models such as Finsformer are widely regarded as black boxes, even though they outperform on predictive tasks. Interpretability is crucial in financial systems as model decisions can have significant economic and legal impacts. Stakeholders such as financial institutions and regulatory bodies often require explanations for model predictions, especially in sensitive areas like credit approvals or fraud detection. Integrating explainable AI (XAI) into Finsformer can enhance the model's trustworthiness and practical applicability.

The reviewed evidence shows that deep learning models help improve phishing and spam detection by learning both local and contextual patterns from text. LSTM models are good at understanding sequences, while self-attention and Transformer-based models focus on better contextual understanding and identifying key features (Xu et al., 2021; Shahariar et al., 2019; Uddin et al., 2026).The results from these studies show that combining different model types works well. CNN helps in extracting local features, and LSTM handles the sequence context. Transformer models

also show the importance of long-range context, but their performance depends on the data and how much computing power is available (An et al., 2024).Overall, the evidence supports using different deep learning methods together for better phishing detection. This study uses a CNN-LSTM model where CNN extracts local patterns from URLs or text, and LSTM captures the bidirectional context within the same system.

Xi. AI-driven detection models

AI-driven models stand to reason that AI-controlled phishing email recognition is an option, as much as it is troublesome due to security issues linked to the playback and reproduction method of testing old systems cannot support (Bauskar, 2024). It can be said to represent a new generation of cyber hacks that demand measures that are equally sophisticated. It ensures that the threat surface is limited so that the organization's weakest links are not as vulnerable, as they can recognize suspicious emails before they are deleted or tagged as spam (Bauskar, 2024). As AI continues to get more advanced, it can be assumed that one of the new solutions is the use of big data Artificial intelligence (AI).

Expanding on this, designers might be viable in combining email datasets with more than email datasets, cross-referencing networks and network latency, and big data may be a solution to discover compromised and controlled devices much earlier, and also produce additional intelligence of the way they work and propagate (Bauskar, 2024).

This notwithstanding, there are also a few challenges with the implementation of the AI-based phishing detection systems based on big data. The systems are intensive in both computational and

sophisticated infrastructure to handle and analyze vast data (Bauskar, 2024). Also, the attackers can strive to circumvent the detection mechanisms by creating phishing messages that resemble the usual patterns of communication, or they can apply methods designed to avoid machine learning models. As such, AI-based detection systems require constant monitoring, training of the model, and updating of the system to make sure that it can still work against emerging cyber threats, thus making it more complicated when detecting phishing threats in emails (Bauskar, 2024).

Phishing detection models based on AI are also evolving into an effective defense tool and a multifaceted technological problem. Although intuitively, the AI-driven phishing email detection is a promising solution, it also creates new challenges, especially in terms of system compatibility, adversarial, and infrastructure requirements. Continuing with the subject matter of the discussion, it is necessary to delve deeper into the implications, opportunities, and limitations of integrating artificial intelligence and big data into phishing detection systems (Bauskar, 2024).

The ability of AI-based phishing detectors to process and analyze large amounts of data in real time is one of the major benefits of this approach. In contrast to the classical rule-based systems, which use pre-established patterns and signatures, AI models, in particular, the models that use machine learning and deep learning, can be trained on previous data and are constantly enhanced to detect new information. This allows organizations to detect phishing attacks that have never been spotted before, including zero-day attacks that would pass through traditional filters. With the help of natural language processing (NLP), such systems can process the semantic meaning of email content, discern unobtrusive linguistic cues, and detect deviations in writing style that can

also signify malicious intent (Bauskar, 2024).

Data availability and quality are also important in the efficiency of AI-based systems for phishing detection. Partial, noisy, or biased data can lead to poor predictions and low model accuracy. For example, a model trained primarily on phishing emails in English will not be able to identify attacks in other languages or cultural settings. This underlines the need to have a variety and representative datasets and effective data preprocessing methods to make model results dependable and unbiased (Bauskar, 2024).

The other significant issue is adversarial behavior. Hackers are also using AI methods themselves to design more advanced phishing attacks. They might involve extremely personal messages produced with the help of sophisticated language models, making them more persuasive and harder to notice. Also, attackers can use evasion methods such as obfuscation, polymorphism, or adversarial inputs to disorient machine learning models (Bauskar, 2024). This generates a continuous arms race between attackers and defenders, in which AI systems must constantly adapt to new and emerging threats.

Interpretability and trust are also problematic with AI-based systems. Most sophisticated architectures, especially deep learning ones, are black boxes, and it is hard to know how a decision is made. This transparency can be problematic in the context of cybersecurity, as organizations may need to understand why some emails are blocked or flagged. Lack of interpretability makes model decisions more difficult to validate, model problems more difficult to troubleshoot, and user trust harder to earn. To solve this problem, it is possible to use explainable AI (XAI) techniques,

which provide insight into the model's behavior and decision-making processes (Bauskar, 2024).

i. Challenges of detecting AI-driven email phishing attacks

Phishing attacks driven by AIs pose a serious problem for conventional detection systems due to their complexity and flexibility (Andriu, 2023). Among the key challenges, it is possible to distinguish:

a. Evading blacklists:

Phishing URLs generated by AI can be modified quickly, challenging blacklist-based detection systems to keep pace (Andriu, 2023).

b. Bypassing heuristics:

AI-driven phishing emails, by imitating valid communication patterns, can evade heuristic-based detection tools that operate under defined rules (Andriu, 2023).

c. Eluding content analysis:

Natural language generation techniques also help attackers generate phishing emails with original, apparently valid content, which is more difficult to detect using content analysis systems (Andriu, 2023).

2.5.2. Metric for email phishing detection Techniques

This proposed study employed common classification metrics to measure detection accuracy and reliability, assessing the effectiveness of a convolutional neural network model for detecting email phishing in cybersecurity using a bidirectional long short-term memory algorithm. Accuracy and

precision determined the email model's ability to label phishing emails without mislabeling legitimate emails, and the percentage labeled correctly was referred to as accuracy.

Recall (sensitivity) as a metric was used to test the effectiveness in detecting actual phishing emails-i.e., decreased odds of phishing attacks being overlooked-and F1-score, which is a harmonic mean of precision and recall. This is most significant in an unbalanced data set, where phishing emails are few compared to legitimate ones. The discriminative power of a set of thresholds will be analyzed using the Receiver Operating Characteristic Area under the Curve (ROC-AUC). The confusion matrices to be filed will include TP, TN, FP, and FN reporting to provide a broader picture of the report. A second group of experts shall also triangulate the quantitative evaluation findings through expert opinion obtained in the form of questionnaires, such that the model may be deemed not only statistically viable but also practical and reliable in the real email environment.

2.5.3. Comparative analysis of the algorithms

The studies that have been reviewed show that Support Vector Machine (SVM), Decision Tree (DT), and K-Nearest Neighbour (KNN) have all been used successfully to detect phishing, spam, and malicious websites. However, the results they report vary depending on the data they used, how they described the features, the way they tested their models, and the measures they used to evaluate them. Because of this, we need to be careful when comparing the accuracy numbers from these studies, since they didn't all use the same data or the same testing conditions.

SVM performed very well, especially when the features were text-based or high-dimensional.

Raza et al. (2021) found around 93% accuracy using the SpamAssassin and PhishTank data sets. Tawil et al. (2024) achieved 94.3% accuracy for SVM in detecting fraudulent emails. Singh et al. (2020) used the Alexa and PhishTank data sets and reported 98.0% accuracy when combining SVM with KNN and Logistic Regression in a hybrid model.

This suggests that SVM works well at separating phishing and real websites when the right features are chosen. However, its performance depends a lot on how the features are chosen, which kernel is used, and how the model's parameters are adjusted. Also, SVM can become very slow when there are a lot of training samples and features.

Decision Tree models have another advantage because they are easier to explain and their classification process is simpler. Raza et al. (2021) found around 96% accuracy in detecting spam. Ahmed et al. (2022) reached 98.80% accuracy for phishing website detection after selecting the right features. Alsariera et al. (2020) reported 97.36% accuracy, an F-measure of 0.974, and an AUC of 0.996 using Random Forest, which is an ensemble method made up of many decision trees. This shows that tree-based models can be very effective when they use the right features. However, traditional decision trees rely a lot on hand-crafted features and may struggle to represent complex patterns in URLs, email texts, or other contextual information.

KNN works in a different way by looking at how similar observations are to each other rather than learning a clear boundary between classes. Its performance is influenced a lot by the number of neighbors (k) selected, the distance calculation used, how the features are scaled, how the classes are distributed, and whether there are any outliers. Raza et al. (2021) and Slam et al.(2022) showed

that KNN can be helpful for phishing and spam detection, but its performance tends to be more sensitive to the data characteristics compared to the stronger tree-based and SVM methods. Slam et al. (2022) reported about 87% accuracy using a hybrid approach with features from CNN-LSTM models and KNN. This suggests that KNN might benefit from meaningful features created by deep learning, but it can still face issues with scalability and class imbalance because it needs to compare each new case with existing training data.

Table 2-1 Comparative analysis of the algorithm

Algorithm	Study	Dataset	Reported Performance	Strengths	Limitations
SVM	Raza et al. (2021)	SpamAssassin and PhishTank data sets	~93%	Effective on text-based and high-dimensional features	Sensitive to feature selection, kernel choice and parameter tuning; slow on large training sets
SVM	Tawil et al. (2024)	Fraudulent email data	94.3%	Effective in detecting fraudulent emails	As above

SVM (hybrid with KNN and Logistic Regression)	Singh et al. (2020)	Alexa and PhishTank data sets	98.0%	Hybrid combination gave the highest accuracy; separates phishing and legitimate websites when suitable features are chosen	Depends on feature choice; computational cost with many samples and features
Decision Tree	Raza et al. (2021)	Spam data	~96%	Easy to interpret; simple classification process	Relies on hand-crafted features; struggles with complex patterns in URLs and email text
Decision Tree	Ahmed et al. (2022)	Phishing website data	98.80%	High accuracy after appropriate feature selection	Dependent on feature selection

Random Forest (ensemble of decision trees)	Alsariera et al. (2020)	Phishing website data	97.36% (F-measure 0.974; AUC 0.996)	Strong ensemble performance with the right features	Relies on hand-crafted features
KNN	Raza et al. (2021)	Phishing and spam data	Not stated	Useful for phishing and spam detection; similarity-based, no explicit class boundary	Sensitive to k, distance metric, feature scaling, class distribution and outliers
KNN (hybrid with CNN-LSTM features)	Slam et al. (2022)	Phishing data	~87%	May benefit from meaningful deep-learning features	Scalability and class imbalance issues; compares each new case with all training data

2.6 Designing an Email Phishing Detection Model

In summary, the technical solution to phishing attacks in cybersecurity involves developing a high-tech email URL-phishing detection model. This model combines Convolutional Neural Networks

(CNNs) and Long Short-Term Memory (LSTM) networks. The CNN component of the model analyzes email URLs, identifying suspicious characters, domains, and URL structures commonly associated with phishing. CNNs have proven effective in detecting phishing emails based on their content, achieving up to 98% accuracy (Atawneh & Aljehani, 2023). The model generates a probability indicating the likelihood that an email is malicious, based on the email body text. CNNs are good at capturing local patterns and n-gram features, which help in identifying hidden phishing words and URLs (Saxe and Berlin, 2017). On the other hand, LSTM networks are better at understanding the flow of information and long-term relationships, allowing them to provide a more contextual interpretation of emails (Phan et al., 2020).

Several studies have combined CNN with LSTM on datasets such as Alexa and Phish Tank, achieving a 98.61% success rate in identifying genuine and phished URLs (Sirigineedi et al., 2021). Wei et al. (2020) used CNN to analyze 73,575 URLs from GitHub and achieved 99.67% accuracy. However, CNNs were limited since they were originally designed for desktop browsers, making it difficult to keep up with rapidly updating lists of trustworthy domains. When Korkmaz et al. (2021) applied CNN to a large dataset of 222,541 URLs, they achieved 88.90% accuracy. However, they faced challenges due to the lack of hybrid algorithms and the use of fewer features.

According to Nanda & Goel (2024), combining CNN and LSTM for phishing detection in emails allows the model to capture both fine-grained local patterns and global contextual features.
The CNN first extracts salient local features that help identify email anomalies, while the LSTM processes these features sequentially to capture temporal and contextual relationships.

2.7 Model Development

The creation of a cybersecurity email URL phishing-detection model based on Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (LSTM) can be structured as follows: data acquisition and preprocessing, model architecture selection, training, testing, and deployment. The objective is to create an intelligent system that detects phishing websites in emails by applying a CNN to extract features and a LSTM within a sequence learning framework. Data collection and preparation that includes gathering a broad range of data consisting of real and phishing URLs and using Kaggle, OpenPhish, and PhishTank websites to obtain the latter is the initial stage of the process. The choice of using datasets like Kaggle, Open Phish, and PhishTank was informed by the fact that they had real, up-to-date, and diverse phishing URLs, many of which were also used in the phishing mails that are currently being common (Rao and Pais, 2020; Sahingoz et al., 2019).

To purify the dataset, null and duplicate values are removed, text is normalized, and major features are filtered. The lexical features (URL length, the presence of special characters, etc.), host-based features (domain age, WHOIS information, etc.), and content-based features (the presence of keywords associated with phishing, etc.) are all factors in feature extraction that need to be considered. Word embeddings such as Word2Vec, GloVe, or FastText are used to encode email content and URLs into numeric representations, enabling the model to learn contextual associations between email content and URL structure (Sahingoz et al., 2019).

Then, CNN and LSTM are combined to form the model architecture. CNNs apply 1D convolutional layers to email content and URLs to identify patterns and obtain spatial and

hierarchical features, using ReLU activation. The dimension reduction using MaxPooling layers does not significantly reduce information. Then the output of the CNN layers is fed into a LSTM, followed by a bidirectional recurrent neural network that models both past and future dependencies in the email message. Such a hybrid approach will ensure that the model identifies both consecutive dependencies in phishing URLs and email bodies (learned by LSTM) and fixed phishing indicators (learned by CNN). Emails are sorted as real or phishing by passing them through features obtained, a fully connected set of layers with a Softmax activation function (Sahingoz et al., 2019).

The training is implemented by dividing the dataset into training and testing sets, usually in a 70:30 ratio. To achieve successful learning, the optimizer will be Adam, and the loss function will be categorical cross-entropy. Accuracy, precision, recall, and F1-score are key performance metrics tracked to achieve robustness. During hyperparameter adjustment, hyperparameters such as convolutional filters, kernel sizes, LSTM units, training batches, and the learning rate are adjusted to improve model performance. To increase model generality and resilience to changing phishing strategies, data augmentation and the generation of dummy phishing URLs are also implemented (Pais, 2020).

After training, the model was tested on test data. The performance was measured using a confusion matrix, ROC-AUC, and precision-recall curves to assess the ability to distinguish phishing from authentic emails. The model was tested on zero-day phishing attacks to assess its flexibility in detecting previously unseen phishing attacks. Lastly, the model also detected phishing immediately. It was turned into a web service using Flask or FastAPI, enabling integration with enterprise security solutions or email security gateways. Real-time threat detection and massive

processing can be achieved by deploying on cloud platforms such as AWS, Azure, or Google Cloud. As phishing tactics evolve, ongoing monitoring of phishing attempts after deployment and retraining with new phishing information would be required to maintain the model's effectiveness.

2.8 Model Validation

Machine learning must critically evaluate the performance of any given Cybersecurity Email Phishing Detection Model, such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM). It is usually done by training the model on labeled emails, both phishing and normal, and testing it on unmarked data to establish performance metrics such as accuracy, precision, recall, and F1 score. The study by Isra Al-Turaiki, Fatimah Alakeel, Reem Alotaibi, and Najwa Altwaijry of King Saud Univ., in which comparative performance was measured between deep learning architectures, that is, CNN and LSTM, was compared to a study by deep learning models to identify phishing emails. The results of the experiment demonstrated that it was possible to obtain an accuracy and F1-score of 99.68 and 99.66, respectively, and compare the results with the model performance of 1D-CNNPD, which allowed the investigator to conclude that the use of recurrent units such as LSTM and Bi-GRU contributed to the detection (Altwaijry et al., 2024).

Besides machine learning validation, expert judgments can also be used to assess the efficiency of machine learning implementation in real-life environments. Cybersecurity experts can provide the facts about the impenetrability of the model to sophisticated phishing attacks and its flexibility to new attacks. The model has been optimally tested and improved to attain a better result with a 2024 paper An Improved Deep Learning Model with a Hybrid LSTM and Bi-GRU Algorithm to Analyze URL and Content Representations in Phishing Detection by Sakthipriya N., Dr. V.

Govindasamy, Abhinesh C., Krishna Govarthini T., Rajesh R., and Swedha R. of Puducherry Technological University (Sakthipriya et al., 2024).

This practice showed the significance of the professional tests in the assurance process and the versatility and effectiveness of the model concerning the protection against changing phishing threats. Phishing detection models based on CNN and LSTM can also be validated comprehensively with machine learning, and expert-based evaluation of phishing can be used to detect or classify phishing correctly. The knowledge of the professionals guarantees the practical issues that are dealt with by the model and its alignment with the existing security plans, but the quantifiable indications of the performance can be identified as an indicator of the performance success. This twofold verification approach is needed to develop an effective and reliable phishing-definition system that can help alleviate the threat of phishing attacks (Sakthipriya et al., 2024).

2.9 Research Gap

Although many methods for detecting phishing already exist, there are still significant research gaps, namely, enhancing the performance of CNN and LSTM models in real-world settings.
One significant drawback of many existing anti-phishing systems is their reliance on blacklist or whitelist-based solutions, which often result in high false positive rates and struggle to detect novel phishing attempts. These traditional methods are limited by their inability to adapt to rapidly evolving phishing techniques, as attackers frequently change URLs and domains Kyaw (2024).

This study builds on earlier phishing detection work by combining URL and text data into one deep learning system called CNN-LSTM. Unlike previous studies that only used manually created

features or standard machine learning methods, this research uses the raw URL or text itself as the main input. This lets the model learn important patterns like meaning, structure, and word order directly from the data instead of relying on features like URL length or number of dots. The study also improves on past methods by mixing two techniques: CNN, which finds small patterns in the URL or text, and LSTM, which understands the full context by looking both forward and backward. The main new part of this study is using all these steps together in one system for phishing detection. This goes beyond previous methods that used separate features, traditional classifiers, or single deep learning models. It shows how both local and sequential patterns can be learned together from URL or text data to help detect phishing more effectively.

The availability of comprehensive datasets for email phishing detection is significantly limited, particularly those concentrating on phishing URLs included within emails. Many researchers have relied on character- level representations to detect phishing threats; however, this method often falls short in capturing the subtle patterns that distinguish phishing emails from legitimate ones (Salloum et al.,2022).

According to Salloum et al.(2022), character-level models are not effective at discovering complex phishing schemes since they only reveal simple patterns, but fail to find the semantic and contextual insights and information. Also, according to Ahmed et al.(2022) attackers often use URL manipulation and email text manipulation that would require more advanced contextual analysis, and character models would miss those; therefore, they are not effective against advanced phishing attacks. To enhance detection, it is recommended that larger and more diversified data should be used, which can help in more successful pattern recognition (Kritika, 2024).

Another notable gap in current phishing detection systems is the challenge posed by phishing emails containing misleading email texts, which are often designed to closely mimic legitimate ones. These altered or truncated links can easily deceive users and evade traditional detection methods. Singh et al. (2020) point out that phishing attackers frequently use domain name spoofing or URL obfuscation to make malicious links appear legitimate, thereby bypassing conventional detection techniques.

Furthermore, Korkmaz et al. (2021) emphasize that many phishing attacks exploit subtle changes in URLs, such as using visually similar characters or shortening the URL, which often go unnoticed by typical security systems that rely on exact matching or simple pattern recognition. As a result, improving the ability to distinguish obfuscated or modified URLs inside email content is critical to reducing the spread of phishing threats and their operational consequences (Zien et al., 2023).

Detecting fine-grained structural elements in different contexts and subsequently extracting them, the hybrid use of CNN-LSTM is a potent solution to email phishing detection that can be highly effective at uncovering both traditional and advanced phishing attempts that are identified in real-time (Rao & Pais, 2020; Sahingoz et al., 2019).

CHAPTER THREE

METHODOLOGY

3.1 Introduction

In this chapter, the author presented details of the method that were employed in the study, such as the research design, which acts as a guideline for performing a study that outlines the design strategy. The research paradigm picked to undertake the research was also encompassed in the research design. The chapter addressed the research design that was embraced as the experimental and qualitative research design to carry out successful identification of phishing threats on the shortened web site URLs and to test the efficacy of the model by the experts. The researcher used the secondary data collection methods for the model development through a dataset. Primary data was used to verify or qualify the ability of the model to solve the problematic threats as tested by the experts.

3.2 Research philosophy

Research philosophy is about the beliefs we have regarding what knowledge is, how the world works, and the best ways to study a problem. This study followed a philosophy called pragmatism because the problem needed both creating a phishing detection model that can be measured and checking if it works well in practice. Pragmatism doesn't force us to choose only between using facts or opinions; instead, it lets us pick the best approach based on what the questions are and what kind of proof we need to answer them.

The reason why pragmatism was chosen it's because the problem was practical. The main goal of the study was to make and test a phishing detection model using CNN-LSTM. To do this, we

needed real-world results showing if the model can tell phishing from real emails. That meant using quantitative methods to check how well the model works with numbers like accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrices. These numbers showed how good the model is technically.

Pragmatism also helped us include a part that looks at how easy the system is to use. Just because a model works well doesn't mean it's useful in real life. So, after showing the system to experts, experts were asked how easy it was to use, how reliable it felt, and how well it fits into real cybersecurity work. This part didn't take the place of the technical tests, but it helped us understand how real users might feel about the model.

Pragmatism affected three key parts of the study. First, it made use of quantitative method to check how accurate the model is. Second, it made us use a survey to get expert opinions on how practical the system is. Third, it let us use both types of evidence together when we analyzed the results, so we could look at both how well the model works and how useful it is in a real setting without one taking over the other.

Pragmatism was not used because of using both types of methods. It was chosen because the problem had two parts: checking if the model can detect phishing effectively and also if the solution is useful in the real world of cybersecurity**.**

3.3 Research paradigm

A research paradigm is the set of beliefs and views that shape how knowledge is created, studied,

and understood .In this study, the research approach was based on two related viewpoints: post-positivism for the part with numbers and experiments, and constructivism for the part with words and user experiences. These different ways of looking at research were used in separate parts of the study, based on what kind of information was needed, not just as different names for the same thing.

3.3.1 Post-positivist Paradigm

The quantitative part of the study was based on a post-positivist approach because the phishing detection model needed to be tested properly using real data. The research didn't rely on opinions from the researchers to say how well the model worked. Instead, it used results from new data that the model had never seen before to measure its performance.

This way of thinking affected how the experiment was set up. The phishing data was split into two parts: one for training the model and another for testing how well it worked on new data. The CNN-LSTM model was trained using the training data and then tested using standard ways to measure classification. Accuracy showed how often the model was correct overall. Precision showed how reliable the predictions were for phishing. Recall showed how well the model found phishing cases. The F1-score balanced precision and recall. ROC-AUC showed how well the model could tell phishing from normal emails. A confusion matrix helped to track how many phishing cases were correctly identified, how many were missed, and how many were wrongly flagged.

The post-positivist approach helped use real, measurable results, controlled experiments, clear steps that could be repeated, and numbers that show how well the model performed.

It also let the researchers compare the CNN-LSTM model to other common machine learning models. The goal was to see if the model actually worked well with new, unseen data.

3.3.2 Constructivist Paradigm

The qualitative part of the study used constructivism because usability and how well the system can be used in real life depend on what people think and feel when they use it. Unlike things like how accurate the model is, usability can't be fully shown by just one number or test. So, experts were asked to share their opinions after using the system or watching it work.

Experts shared their views on how easy the system was to use, how reliable it was, how confident they felt using it, whether it could be used in real life, and how well it worked with other tools. These opinions helped us understand how real users might feel about the system. It's important to note that the questionnaire wasn't used to check if the model was technically correct. That part was handled through other tests described in the post-positivist part of the study.

3.4 Research design

A research design was used as a “structure to determine, among other things, how to gather additional data, analyze and interpret them, and ultimately, to provide an answer to the question" (Rezigalla, 2020). The study addressed the first objective, focusing on experimental design to evaluate the effectiveness of existing phishing anomaly detection models in emails to identify parameters for improving the proposed models. The purpose of the experimental research design in objective one was to compare the effectiveness of existing phishing detection models and to identify the key parameters or attributes used in the development of the model.

To effectively identify phishing URLs and emails texts, a combination phishing-sinking model was developed that combined Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models. This model adopted CNN feature extraction capabilities to extract local patterns within email, and URL structures, and LSTM was adopted to evaluate sequential dependencies and contextual connections on the information (Ahmed et al., 2022).The model was built using either a developmental research design focused on designing an effective CNN-LSTM model to address phishing challenges, or an experimental design examining the dynamics among the model variables and how they affect the model's detection performance (Kritika, 2024).

The paper followed a qualitative research design in validating a convolutional neural network-based model in detecting phishing. Questionnaires were used to support the main aim of the research, which was to ascertain whether the given model can effectively solve the problem of phishing by incorporating professionals. In adopting the experimental research was to ensure that the model is created to identify the threats affecting the operations of the pages through phishing on URLs by creating a desktop application that identifies the threat in real time on URLs (Kritika, 2024).

The study adopted an experimental research design for model development and evaluation, and a qualitative design for usability assessment. The model achieved 97.88% accuracy based on experimental evaluation metrics, while experts assessed system usability through questionnaires.

3.5 Model Development Procedure

The following steps were followed in the model development.

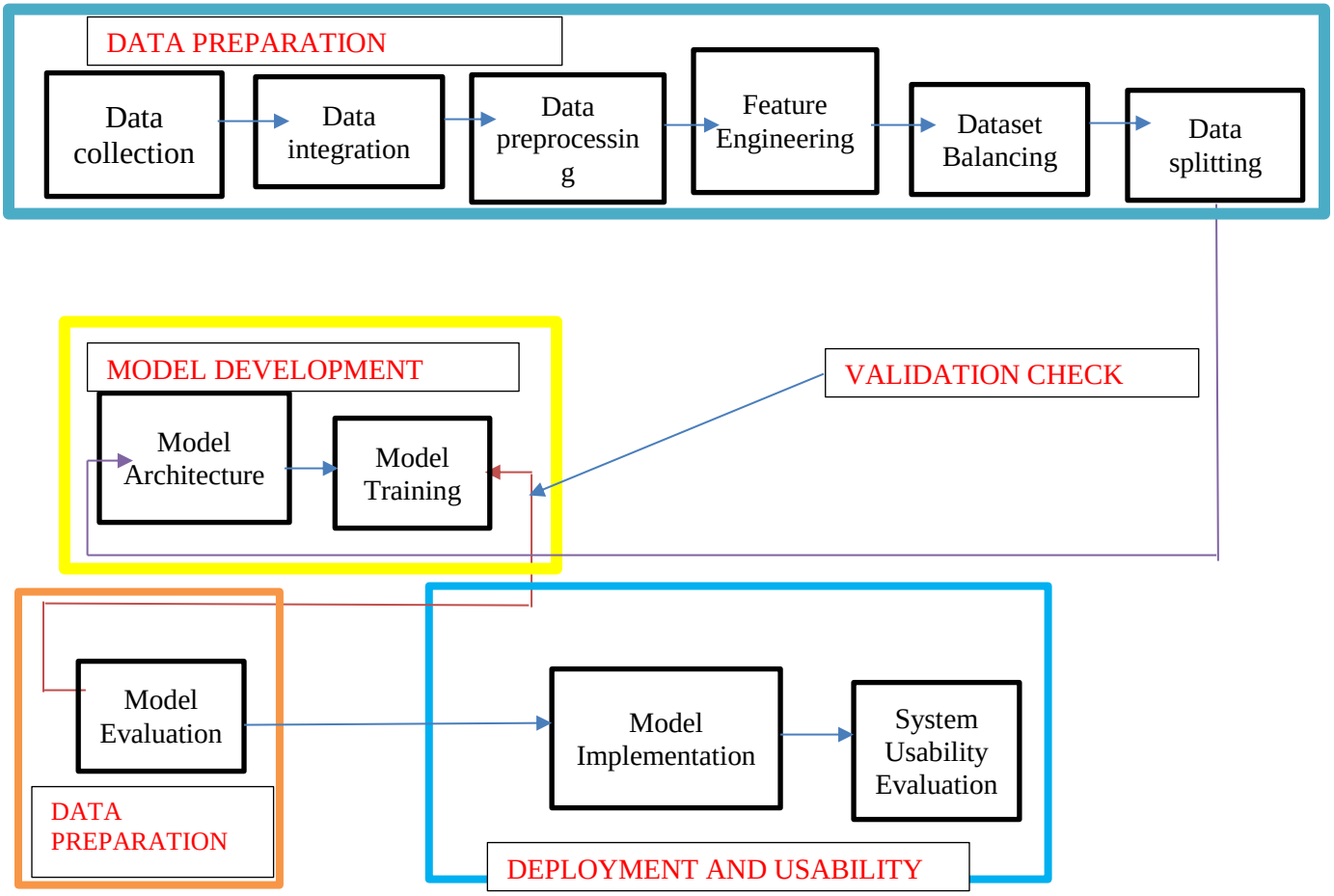

Figure 3-1 Model Development Procedure Flowchart

3.5.1 Data Collection

In this work, pre-collected data from reliable sources have been used for the model development purposes. In particular, the work uses the Phishing and Legitimate Emails dataset available on Kaggle (https://www.kaggle.com/datasets/kuladeep19/phishing-and-legitimate-emails-dataset1). The use of this dataset is appropriate due to the considerable number of labeled emails in it which makes it convenient for both model training and evaluation.

In the initial dataset, there were 10,000 emails after the removal of duplicates and invalid observations, 8,986 emails remained. The data were categorized in phishing and legitimate categories. Data were presented in CSV format, and it included actual emails together with the label value 1 in case of phishing email and 0 for legitimate emails. The use of data from one source helped to preserve consistency in data structure, labels, and features. In total, two stages of research were developed in the current study: development of the model and testing of the model.

Development of the model involved preparation of the data through its cleansing, its splitting into parts, and equalization of the emails' lengths. Then, the architecture of the model was designed, and the process of training began. About 70 percent of the dataset (that is 6,290 observations) was used for training. Some part of the training data was used for evaluation of the model performance and selection of the optimal values of hyperparameters like the number of training iterations and number of filters.

Testing of the model implied evaluation of the performance of the model based on data that was not used for training. It accounted for 30 percent of the data, that is, about 2,696 observations. The model performance was estimated with the help of accuracy, precision, recall, F1-score, ROC-AUC and confusion matrix. During the testing stage, the model was compared with some simple models K-Nearest Neighbors and Random Forest. Test results were analyzed statistically and evaluated by 20 domain experts.

Before developing the model, the dataset was validated for its completeness and appropriateness for the research. The consistent nature of the dataset helped to facilitate the processing of the data,

representation of features and classification.

3.5.2 Data Integration

The dataset was first gathered in CSV format. A clear method was followed to combine the dataset, ensuring it was consistent, high-quality, and ready for use in machine learning models for detecting phishing emails. During this process, both the text of the emails and their classification as either phishing or legitimate were brought together into one format. The text of each email served as the input data, while the classification labels showed whether the email was phishing (marked as 1) or genuine (marked as 0).

The datasets were combined using the Python Pandas library. Each file was loaded into a single DataFrame, matched to the defined structure, and merged into one dataset. This made the data uniform and properly organized for further analysis. In the end, the dataset was clean, standardized, and prepared for extracting features, training, validating, and testing the proposed Convolutional Neural Network (CNN) model. The following code in Figure 3.2 shows how the data was integrated.

```
# DETECT TEXT + LABEL COLUMNS
# ============================================================
text_col = next(
    c for c in df.columns
   if "text" in c or "email" in c or "message" in c
)

label_col = next(
    c for c in df.columns
    if "label" in c or "class" in c or "target" in c
)
```

Figure 3.-2 Dataset integration

3.5.3: Data Preprocessing

Data preprocessing was done to improve the quality of the data and help the model work better in detecting phishing emails. The process went through several important steps to turn raw text into a clean and organized format that could be used for deep learning analysis.

First, duplicate and unnecessary records were removed to avoid bias and ensure the data was unique. Any incomplete or missing information was dealt with by either deleting those parts or filling in the missing data where it made sense, so the data stayed accurate and reliable as shown in Figure 3.3 below.

```
# remove near duplicates
df[text_col] = df[text_col].str[:250]

df = df.drop_duplicates(subset=[text_col])
```

Figure 3-3 Removing duplicates

Next, text normalization methods were used. This included changing all text to lowercase, removing unwanted characters, and stripping out things like protocol names and extra symbols. This helped make all the text the same format and easier to work with. After that, the cleaned text was split into smaller parts called tokens, which are like individual words or meaningful units. This made it easier for machine learning models to understand and use the data effectively.

```
# ============================================================
# TRAIN / TEST SPLIT
# ============================================================
X_train_text, X_test_text, y_train, y_test = train_test_split(

    X_text,
    y,

    test_size=0.30,

    random_state=42,

    stratify=y
)
```

Figure 3-4 Text Splitting

Finally, since deep learning models cannot work with text directly, the processed text was turned into numbers using techniques like token encoding and word embeddings, such as Word2Vec or embedding layers. This step allowed the model to learn and understand the meaning and relationships between words in the data.

3.5.4 Feature Engineering

To help the model better tell the difference between phishing emails and real ones, feature engineering was used to find both obvious and hidden patterns in the text. Two main types of features were used. The first type was structural features, which looked at things like how long the email was, how often it used punctuation like dots or special symbols such as @, -, and _, whether it had suspicious parts like links, and if it included words often found in phishing emails like "login" or "verify." The second type was learned features, which were created automatically through the deep learning model's ability to understand word meanings and how words relate to each other in the email. By combining these carefully made structural features with the model's automatically learned ones, the model's ability to detect phishing emails improved a lot.

3.5.5. Dataset Balancing

The random under-sampling technique was used to address class imbalance and avoid bias towards the majority class. The dataset was balanced by ensuring an equal number of phishing and legitimate emails, with approximately 4493 samples in each category. This balanced representation ensured that the model was exposed equally to both classes during training, reducing the chance of biased predictions that favor one class over the other. Class imbalance is common in cybersecurity datasets, where legitimate emails usually make up the majority, leading to skewed learning outcomes. By balancing the classes, the model was able to learn more distinct patterns from both phishing and legitimate emails, resulting in more reliable, stable, and fair classification performance.

3.5.6. Data Splitting

The dataset was divided into training and testing parts in a 70:30 ratio. 70% of the data, 6,290 samples from the emails dataset, was used to train the model. The rest, 2696 samples, were used for testing and evaluation. The training data helped the model recognize patterns and differences between phishing emails and real ones. The testing data checked how well the model could handle new, unknown emails. This method ensured that the evaluation showed how well the model could perform on new data, giving a fair and realistic idea of its performance in practice and avoiding skewed learning outcomes. By balancing the classes, the model learned more distinct patterns from both phishing and legitimate emails, resulting in more reliable, stable, and fair classification performance.

3.5.7 Model Architecture

The model was implemented using Python (TensorFlow/Keras). Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (LSTM) were used together in the developed model to leverage their complementary strengths in detecting phishing emails. The CNN component extracted local patterns in the email text, enabling the model to identify important substrings, structural patterns, and hidden features typically associated with phishing attempts. This was followed by LSTM, which was used to encode sequential information in the email datasets by processing the data in both forward and reverse directions, thereby enabling the model to capture contextual connections and patterns across the complete sequence of emails. The last step involved the classification task of each input as phishing or legitimate by a fully connected (dense) layer. This combination architecture was very useful as it was highly successful in increasing detection accuracy because the combination of CNN feature extraction and LSTM sequence learning capabilities gave rise to a stronger and more trustworthy phishing detection model.

3.5.8. Model Training

The model was trained using a supervised learning method, where data that was already labeled was used to help the model learn how to classify emails as either phishing or legitimate. The Binary Cross-Entropy loss function was chosen because it works well for tasks like this, helping the model reduce the difference between what it predicts and the actual results. The Adam optimizer was used to make the training process more efficient and to adjust the model's settings in a smart way, which helped it learn faster and more reliably. The following snippet on Figure 3.5 shows the use of Adams optimizer.

```
model.compile(
    optimizer="adam",
    loss="binary_crossentropy",
    metrics=["accuracy"]
)
```

Figure 3.5 Model optimization

The training setup was carefully adjusted by changing the number of epochs and batch size to find the best balance between the models not learning enough and learning too much.

As the training went on, the model gradually picked up on key differences between phishing and regular emails. During this time, the performance of the model was regularly checked on new data to make sure it could handle situations it had not seen before.

The model ran through five training rounds, and each round showed steady improvement.

The accuracy started at 59.57% in the first round and rose to 67.88%, 73.08%, 77.27%, and finally reached 81.20% in the fifth round. Each training round had 148 steps, meaning the dataset was split into 148 groups for each round. Throughout the training, the model's performance on the validation data was closely monitored to ensure it would perform well on new and unseen emails.

3.5.9. Model Evaluation

The model was tested using numerical performance measures to provide a clear and thorough evaluation of its ability to detect phishing emails. Common classification metrics, including accuracy, precision, recall, and F1-score, were used to assess how well the model performs overall, how many emails it correctly identifies, how well it spots phishing emails, and how well it balances catching real threats without too many false alarms.

Additionally, a confusion matrix was used to provide a breakdown of true positives, true negatives, false positives, and false negatives, allowing a closer look at how the model's predictions compare to the actual results. The Receiver Operating Characteristic (ROC) curve and the Area under the Curve (AUC) were also used to assess how well the model distinguished between phishing and non-phishing emails across various thresholds. Together, these evaluation methods provided a thorough and unbiased assessment of the model's performance and classification skills.

3.5.9 Model Implementation

The model was tested in a desktop-based application to demonstrate its practicality in real-world scenarios. The system was designed so that the user pasted the links and text, which were then processed by the trained CNN–LSTM model. The model used characteristics that were learned from the text in emails after some initial processing and turning the text into numbers. These characteristics include features at the word level, created using pre-trained GloVe Word2Vec embeddings. These embeddings help capture the meaning of words related to phishing, such as words that signal urgency, financial terms, requests for authentication, and expressions related to security. The final numerical representations were adjusted using a scaling method to make sure all the features were on the same scale. This implementation supported the idea that the model was not only theoretical but also practically deployable, making it suitable for real-world cybersecurity environments and effective at detecting phishing. The output generated was created using Flask API.

Step 1: pasting an Email to a Flask API as shown in Figure 3.6

The step involved pasting the email address on the interface to allow a user to analyse the email. The text, URLs and Links pasted on the interface involved the phished text or legitimate emails to test the nature of the emails as shown below.

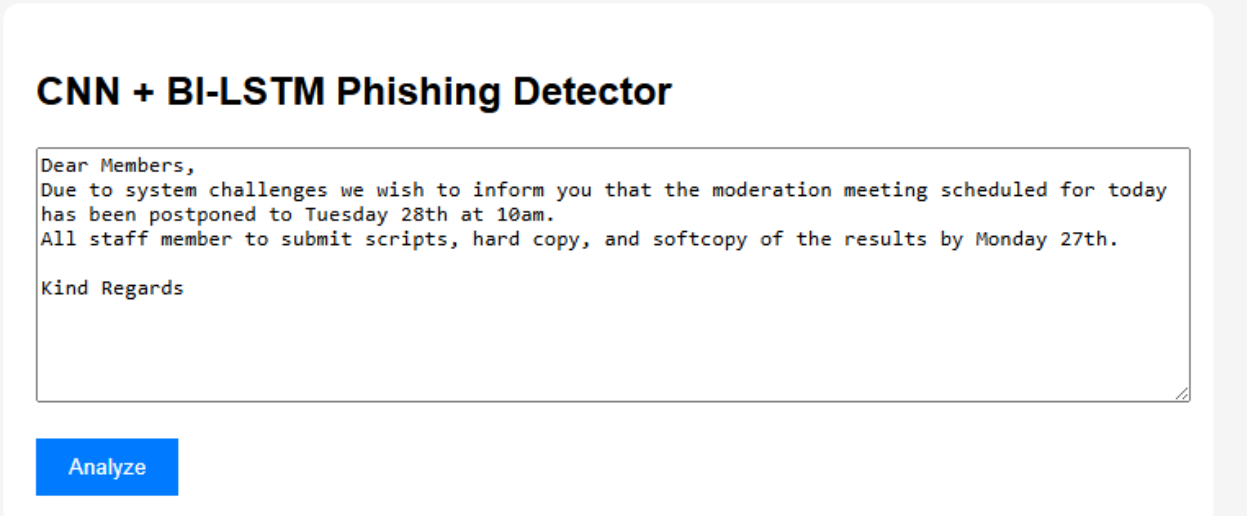

Figure 3.6 Working with Flask API

Step2: Analyzing the model as shown on Figure 3.7

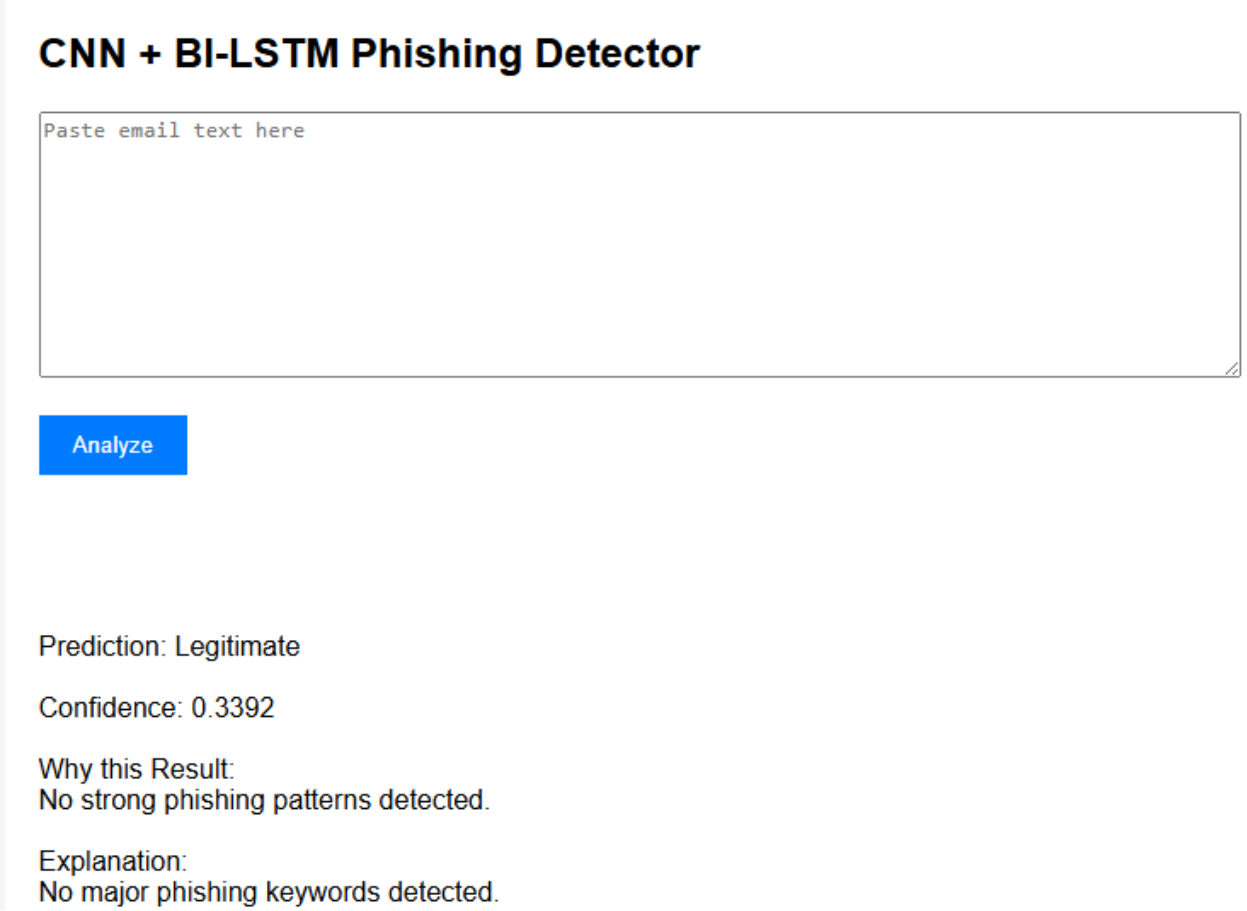

Figure 3.6 Analyzing the model

The model, after analyzing indicated that the email does not exhibit strong characteristics commonly associated with phishing attacks, with a confidence value of 0.3392 which indicated 33.92% chances of phishing, which is below a typical threshold of 0.5 and therefore classified as legitimate email. Since the confidence is close to Zero (0), the model is not highly certain, which indicates some weak suspicious patterns which are insufficient to classify the email as phishing.

3.5.10 System Usability Evaluation

The questionnaires, in contrast to performance evaluation, were used to assess the usability of the developed system. A team of cybersecurity specialists were involved to assess the main features of the system, including its ease of use, the efficiency of the interface design, its convenient applicability, and elements of its relevance in real-life phishing detection situations. This qualitative analysis was a supplement to the quantitative findings that were acquired through performance measures to be sure that, besides the fact that the system is performing correctly, it is also fulfilling user and operational demands in real-life settings.

3.6 Positive predictive value of the Model

Positive Predictive Value (PPV), also called precision, was used to check how reliable the CNN-LSTM phishing detection model is in identifying phishing emails. Positive predictive value shows the percentage of emails that the model says are phishing and are actually phishing. This tells us how trustworthy the model is when it labels an email as phishing. A high Positive predictive value, means the model makes few wrong positive predictions, meaning it rarely says an email is phishing when it is not.

The Positive predictive value, was especially important in this study because it helped find out how accurate the phishing predictions made by the CNN-LSTM model were.

For example, if the PPV was 0.96, it means about 96% of the emails the model labeled as phishing were truly phishing, while about 4% were false positives under the testing conditions.

Positive predictive value**,** showed how precise the model was in its positive phishing predictions and worked together with recall, which measured how well the model found all the phishing emails

in the test data.

Using Positive predictive value allowed the study to not only check if phishing emails were found, but also to see how reliable the model's phishing predictions were.

3.7 Negative predictive value of the Model

A high Negative Predictive Value means that most emails the model marked as safe were truly safe. On the other hand, a lower Negative Predictive Value means more of the emails marked as safe were actually phishing emails. This is especially important in phishing detection because if the model wrongly marks a phishing email as safe, it might reach the person who is supposed to get it.

For example, a Negative Predictive Value of 0.95 means that 95% of the emails the model said were safe were really safe, while about 5% were incorrect safe classifications.

Negative Predictive Value helps show how reliable the model is when it says something is safe, and it checks the opposite of what Positive Predictive Value does.

Using both PPV and NPV together gives a better idea of how well the model works. PPV checks how well the model finds phishing emails, while Negative Predictive Value checks how well it identifies real emails. Along with accuracy, recall, specificity, F1-score, and ROC-AUC, these measures give a clearer picture of how good the CNN-LSTM model is at classifying emails.

3.8 Research population

The study comprised all potential phishing Emails and legitimate Emails alongside their

characteristics originating from different sources. The research population which consisted of the following:

3.8.1 Target population

The dataset was made up of two groups from the emails dataset: phishing emails and legitimate emails. Phishing emails were fake or harmful messages meant to trick users, usually trying to get sensitive information or lead them to dangerous websites. On the other hand, legitimate emails were genuine and trustworthy messages sent by verified sources like reliable organizations, official services, and secure websites. This clear difference between the two types helped the model learn important patterns better, making it more accurate at telling apart phishing emails from real ones.

The study included 20 experts in the field who had been qualified for the task for the validation of the model. Since there were only a few people who met these qualifications of an expertise in cyber security, all 20 of the experts were included in the research. So, the sample size of the study was n = 20, which represented 100% of the people were to be studied. Using the whole population of experts was the correct choice for the research since omitting any of the experts would have resulted in a loss of information regarding the system that was being developed.

3.8.2 Population Size

The number of emails used in the research was based on the study's goals. A total of 10,000 emails dataset were gathered from the dataset then binning was done and the actual dataset that was left was 8886 which was divided into two groups: one for training the model and another for testing. The split was done in a 70:30 ratio, meaning 70% of the emails were used to train the model, while

the remaining 30% were kept aside for testing and checking how well the model works. This way, the model could learn from a big part of the data and still be tested on emails it hadn't seen before, giving a good idea of how well it performs in real situations.

3.9 Ethical issues

Research ethics are some of the genres that researchers adhere to in order to safeguard participants' rights when creating research techniques and establishing a reliable rapport between the researcher and study participants. Research ethics, which refers to an ongoing experience of creating new knowledge for the purpose of revealing a fresh stream of ideas in academia, is highly important in scientific studies (Rana et al., 2021).

The researcher described the study's goal and significance to the institution to obtain permission to collect data. To guarantee that the information gathered from respondents is secure, unhindered, and shielded from unauthorized use, privacy and confidentiality measures were implemented. The researcher gained knowledge on the confidentiality of the dataset since it included sensitive and private data, which when misused, created immense loss in the access of the online data. This was adhered to by ensuring that there is compliance with data protection laws which included general data protections regulations (data protection act 2019 in Kenya, Nigeria data protection regulations 2019 and data protection law and regulation in USA 2024 etc.).

The study considered the concerns of bias in training data which leads to unequal treatment of URLs from certain region or domain instead of considering a distributed training of data. The researcher considered a diverse representation of dataset during training to promote fairness in

model evaluation process. The researcher took into account the ability to regulate the misuse of technology because most of the algorithms could be applied to avoid detection, which undermined operations and made it difficult to detect phishing threats. The researcher also noted the use of guidelines in the application of the technology as well as limited access that was used on the model.

3.10 Conclusion

The research methodology described in this chapter laid the groundwork for the creation and testing of a cybersecurity phishing detection model that employed Convolutional Neural Networks (CNN) in conjunction with Bidirectional Long Short-Term Memory (LSTM) networks to specifically target email URLs. The study successfully fulfilled its aims by using a mixed-methods strategy that incorporated both experimental and qualitative research designs. The experimental approach allowed for a structured analysis of existing phishing detection systems, which aided in the identification of crucial parameters to improve CNN-LSTM model. Furthermore, the model's efficiency was evaluated using qualitative approaches, such as expert input obtained through questionnaires, to ensure both technological robustness and practical application (Asiri et al., 2023).

The study combined primary and secondary data, with the former being used to test the model's performance and the latter providing a solid training dataset that was obtained from reliable sources, which included Kaggle dataset. A balanced dataset of 8986 emails, with 70% designated for training and 30% for testing, guarantees an impartial and exacting evaluation of the predicted accuracy of the model (Opara et al., 2019; PhishRepo Consortium, 2022).

Ethical issues were carefully taken into account, with a focus on data privacy, adherence to regulatory requirements, and impartiality in dataset portrayal to avoid prejudice. Through the implementation of stringent usage restrictions and access control systems, the study also protected against the misuse of the produced Technology.

The research aimed to develop a novel, trustworthy, and useful phishing detection model that could address the evolving challenges of cybersecurity threats using this methodical approach.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

This chapter offers a thorough analysis and explanation of the results pertaining to the methods employed in phishing detection research. Analyzing the existing phishing attacks detection techniques to identify techniques /technology used, strengths, weaknesses, and security loopholes. From conventional classifiers like Support Vector Machines (SVM) to sophisticated hybrid models like CNN-LSTM, the chapter explores how well these methods work in identifying phishing threats across various datasets. Building on this study, the next stage is to create an effective phishing detection model based on Convolutional Neural Networks (CNNs) that integrates the best practices found in the assessed methods. Because CNNs can automatically extract high-level characteristics from input data, including email content, URLs, and header information, they are especially well-suited for text-based categorization tasks.

Building on this research, the next step was to develop a Convolutional Neural Network (CNN)-based phishing detection model that incorporated the best practices found in the evaluated techniques. Because CNNs can automatically extract high-level features from both structured and unstructured input data, including email content, embedded URLs, and header metadata, they are especially well-suited for text-based categorization tasks. The suggested model will be set up to handle major phishing detection issues, like high false-positive rates and flexibility in response to changing phishing trends. Lastly, the model will be validated to evaluate its robustness, accuracy, and dependability. The model's efficacy in practical situations and its capacity to proactively counteract phishing threats will be ascertained through this validation approach.

4.2 OBJECTIVE 1: Analysis of Existing techniques used in phishing attack detection.

The table below shows the results from 28 journals that were chosen out of 40 found through Google Scholar. It gives an overview of the different algorithms used in those studies as shown on the Table 4.1 below.

Table 4-1 Selected Journals

Author & Title of Journal & Year	**Technique / Algorithm Used**	**Dataset**	**Strengths**	**Weaknesses / Security Loophole**	**Accuracy (%)**
Madhavaram et al. (2023).	Support Vector Machine	PhishTank, Public Dataset	Good generalization, effective with high-dimensional data	Sensitive to noisy data	95.6
Raza et al. (2021).	Support Vector Machine	SpamAssassin, PhishTank	Efficient classification	May not scale well with large datasets	93
Ahmed et al. (2022).	Support Vector Machine	Twitter, Facebook	Handles unstructured social data	Social media data may have biases	-
Zuraiq & Alkasassbeh (2019).	SVM + Naïve Bayes	PhishTank	Combined strengths improve classification	Naïve Bayes assumes independence	-
Singh et al. (2020).	SVM + KNN + Logistic Regression	Alexa, PhishTank	Ensemble increases accuracy and robustness	Computationally expensive	98
Raza et al. (2021).	Decision Tree	UCI or custom phishing email dataset	Easy to interpret, fast	Overfitting with complex datasets	96
Alsariera et al. (2020).	Random Forest (DT Ensemble)	UCI phishing dataset	High accuracy, handles overfitting	Slower than single trees	97.36
Alsariera et al. (2020).	PWCAC (DT-based)	UCI phishing dataset	Adaptive decision mechanism	Might lack interpretability	-

Taloba & Ismail (2019)	GADT (DT + Genetic Algorithm)	Custom	Optimizes feature selection	Complexity in tuning genetic parameters	-
Slam et al. (2022).	K-Nearest Neighbors	UCI + Synthetic	Simple, non-parametric	Slow with large datasets	87
Safi & Singh (2023)	Random Forest	Kaggle URLs	Effective with URL-based features	May not perform well on email body text	97
Safi & Singh (2023)	Neural Network	GitHub Dataset	Learns deep representations	Requires large amounts of data and time	96.6
Zabihimayvan & Doran (2019)	Neural Network	PhishTank, Miller Smiles	Flexible for various data types	Overfitting if not regularized	-
Atawneh & Aljehani (2023)	Feedforward Neural Network	Kaggle URLs	Fast training with simple architecture	Limited memory of sequence data	93
Ahmed et al. (2022).	Fuzzy DT + Naïve Bayes	Spam emails (1000)	Handles uncertainty in decision-making	Less scalable	-
Zuraiq & Alkasassbeh (2019).	Neuro-Fuzzy Scheme	Custom	Good at capturing uncertain and fuzzy features	Computational cost	98.5
Salloum et al. (2022).	NLP (Review)	Multiple	Comprehensive language understanding	Not specific to phishing classification	-
Adebowale et al. (2023).	LSTM	Phishing URL dataset	Captures sequential patterns in URLs	Long training time	-
Sahingoz (2018)	ANN/DNN	Custom	Learns complex patterns	Overfitting without enough data	-
Adebowale et al. (2023)	CNN + LSTM	Phishing Websites	Combines spatial and temporal features	Computationally intensive	-
Rathee & Mann (2022)	CNN	Custom	Effective with spatial patterns in text	Ignores sequence if not combined with RNN	95.97
Peng et al. (2019)	CNN + LSTM + Attention	Phishing URLs	Focus on important features	Model complexity	98.26

Feng et al. (2018)	CNN + Word2Vec	Sina Weibo	Embedding improves feature understanding	May struggle with sarcasm or informal text	91.36
Seth & Biswas (2017)	CNN + Text + Image	Custom	Multimodal detection	Requires image preprocessing	98.11
Shahariar et al. (2019)	CNN + LSTM + Word2Vec	Custom	Strong semantic and sequential feature capturing	Computationally expensive	94.56
Wang et al. (2021)	CNN + LSTM	PhishTank, SpamAssassin	Handles both spatial and temporal features	Model size	97.8
Xu et al. (2021)	LSTM + Self-Attention	Twitter, Weibo	Selective focus on important parts of sequence	Complex tuning	91
Shahariar et al. (2019)	LSTM + CNN + Word2Vec	Custom	Balanced feature extraction	Resource intensive	95
Kyaw (2024), Applying Deep Learning for Detecting Phishing on Emails	CNN, SVM, LSTM, LSTM, CNN-LSTM	Email phishing dataset	Highly flexible in learning patterns, structures, and features. The hybrid CNN-LSTM achieved the highest performance.	NLP showed poor results due to language limitations; it was not effective for dynamic email threats.	CNN-LSTM: 99.42% SVM: High NLP: 2%
Alshingiti (2023), Website Phishing Detection	CNN, LSTM, LSTM-CNN	Web-based phishing dataset	CNN is robust in extracting features; LSTM-CNN and LSTM models handle sequential data well.	May struggle with semantic complexity and advanced obfuscation.	CNN: 99.2% LSTM-CNN: 97.6% LSTM: 96.8%
Roy et al. (2022). Malicious vs	LSTM, LSTM	URL datasets (malicious and benign)	LSTM processes sequence data bidirectionally	Might be resource-intensive and	LSTM: 97% LSTM: 99%

Benign URLs Detection			; it excels at detecting contextual patterns in URLs.	sensitive to input structure.	
Meléndez (2024), Email Phishing Detection with Traditional and Transformer-based Models	Logistic Regression, XLNet, BERT	Email dataset	Transformer models (BERT, XLNet) had high accuracy.	Logistic Regression performed poorly due to misclassification of grammar errors, leetspeak, and HTML content.	Logistic Regression : 2% BERT: 99.1% XLNet: 98.84%
Thakur (2023), Comparison of BERT and Word2Vec in Phishing Email Detection	BERT, Word2Vec	Email phishing dataset	Word2Vec outperformed BERT when used with feature selection techniques.	BERT underperformed with Chi-square feature extraction; it is not ideal with limited input preprocessing.	BERT: 98.2% Word2Vec : 98.8%

Table 4-2 Techniques Distribution

Algorithm	Frequencies
CNN	7
SVM	6
LSTM	5
Decision Tree (DT)	4
LSTM	4
NN / ANN / DNN	4
Naïve Bayes	3
Word2Vec	3
Random Forest	2
K-Nearest Neighbor (KNN)	2
Self-Attention	2
Fuzzy Decision Tree	2

Logistic Regression	1
Transformer (ALBERT)	1
Text/Image Input	1
NLP	1

Table 4.2 above shows the frequency of the various algorithms used by different authors in phishing detection models for email attacks using supervised learning.

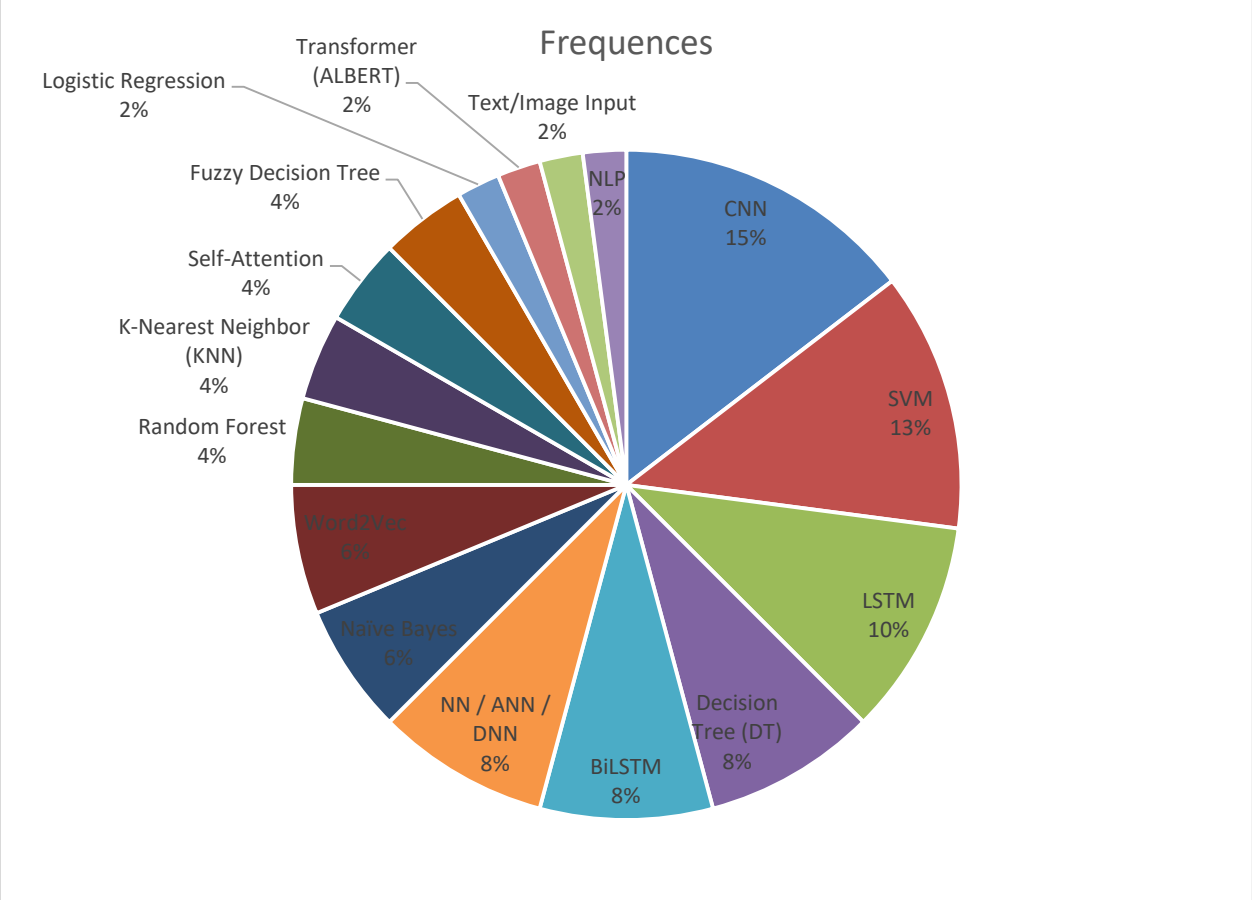

Figure 4-1 Algorithm Used

4.2.2 DISCUSSION

According to Figure 4.1, the comparative analysis of various machine learning and deep learning models for detecting phishing emails reveals noticeable variation in performance, accuracy, and adaptability to complex data structures. The Convolutional Neural Network (CNN) emerged as the

most effective model with a score of 15%, followed by the Support Vector Machine (SVM) at 13%, and the Long Short-Term Memory (LSTM) network at 10%. The remaining models, including Natural Language Processing (NLP), Logistic Regression, and a CNN variant that accepts both text and image inputs, scored 2%. These performance disparities highlight the influence of the model architecture, data encoding, and learning mechanisms on the detection capabilities, especially in distinguishing phishing emails from legitimate ones. In his study on the application of deep learning to phishing detection, Kyaw (2024) states that deep learning models, including CNN, LSTM, and LSTM, demonstrate the ability to effectively, specifically, and automatically learn textual and structural features of emails, leading to improved detection outcomes compared to other machine learning algorithms.

The superior performance of CNN in this study can be attributed to its capacity to capture local patterns in the input data. Though originally designed for image classification, CNNs have shown remarkable adaptability in text-based phishing detection. Their convolutional layers can effectively extract n-grams, word embeddings, and local patterns from email text, subject lines, and URLs. CNNs can detect subtle differences and associations indicative of phishing intent, such as abnormal lexical syntax, suspicious URL tokens, or the presence of psychologically manipulative keywords. The use of convolution and pooling allows CNNs to filter out noise, highlight important features, and generalize well, even when facing previously unseen phishing formats. Additionally, CNNs have shorter training times than recurrent networks like LSTMs, as convolutional filters can be processed in parallel, making them computationally efficient without sacrificing high precision. This efficiency likely contributed to the highest score of 15% for CNN, supporting Kyaw's (2024) findings that CNNs are the most resilient among deep learning

classifiers in phishing email detection.

The second-best model, the Support Vector Machine (SVM), achieved 13% accuracy and demonstrated strong performance, reaffirming its relevance as a classical machine learning approach for binary classification tasks such as distinguishing phishing emails from legitimate ones. SVMs are effective because they learn optimal hyperplanes that maximize the margin between data points of different classes. In phishing detection, SVMs successfully classify structured features extracted from email metadata, URLs, and content, such as word frequency and character distribution. However, their performance is slightly lower than that of CNNs, suggesting that while SVMs excel at separating non-linearly separable datasets using kernel tricks, they may struggle with the hierarchical and sequential nature of natural language data, a task better handled by deep learning models like CNNs and LSTMs. Deep learning models inherently capture such contextual information, whereas SVMs require extensive manual feature engineering and preprocessing. Despite this, the relatively high score of SVM indicates its potential to match deep learning models' accuracy when combined with appropriate feature extraction strategies, such as TF-IDF, n-grams, or word2Vec embeddings. It is particularly useful in resource-constrained environments or with smaller datasets.

The Long Short-Term Memory (LSTM) model, scoring 10%, performed well but slightly worse than CNN and SVM. As a variant of the recurrent neural network (RNN), LSTMs are designed to learn long-term dependencies in sequential data through memory cells and gating mechanisms, making them suitable for detecting phishing activities that rely on contextual information in text. For instance, phrases like "verify your account," "update your password," or "urgent action

required" can be used in phishing attempts, but are usually harmless in formal company communications. LSTMs can distinguish between such cases by analyzing the context in which these phrases appear. However, LSTMs come with several limitations, including computational intensity, a tendency to overfit, especially on small or imbalanced datasets, and longer training times due to their sequential processing nature. These factors may explain why CNN outperformed LSTM in this analysis. Despite these drawbacks, according to Kyaw (2024), LSTMs and their time-aware counterpart, LSTM, are valuable for capturing contextual semantics and time-related relationships that other models may not account for.

At the bottom of the performance table were Natural Language Processing (NLP)-based traditional models, Logistic Regression, and CNNs trained on mixed text/image data, all scoring 2%. The poor performance of standalone NLP models may stem from their reliance on handcrafted linguistic features such as token frequencies, part-of-speech tags, and sentiment predictions, which are ineffective at capturing the multidimensional and dynamic nature of phishing emails. Unlike deep learning models, traditional NLP methods do not learn representations; instead, they rely on fixed rules or small statistical models that may not be effective at detecting new phishing techniques. Logistic Regression, for instance, performed poorly for similar reasons. While it is efficient and interpretable, Logistic Regression assumes a linear relationship between features and the target class, which is often invalid in phishing detection problems involving nonlinear, high-dimensional data. Moreover, Logistic Regression is not very effective with text features represented as sparse vectors, such as those created by TF-IDF or Bag-of-Words, leading to poor generalization when faced with varied email formats or misleading linguistic signals.

The poor results of CNN models trained on text and image inputs (2%) are also intriguing and require further interpretation. Primarily, multimodal methods, which analyze both textual and visual information (e.g., logos, in-text imagery, or brand templates), should be more effective at distinguishing between authentic and fake phishing sites because they provide additional information. However, there are challenges associated with this approach. To begin with, the image data used in phishing emails can vary in quality, resolution, and encoding, making it difficult for CNNs to extract meaningful visual features. Additionally, the fusion of image and text representations involves complex network structures, and balanced datasets may prevent either modality from overpowering the other. If the text already contains sufficient discriminative information, adding image features may introduce noise rather than improve accuracy. Therefore, the poor performance of CNN models on text/image inputs may be attributed to data imbalance, low image quality, or insufficient multimodal fusion during training.

All these results support a key finding in phishing email detection research: deep learning algorithms significantly outperform traditional machine learning algorithms because they can learn and generalize from large-scale data without explicit feature engineering.

Specifically, CNNs remain the top-performing models in terms of accuracy, computational efficiency, and robustness to noise. This conclusion is also supported by Kyaw (2024), who emphasizes that deep learning architectures, particularly CNN, LSTM, and LSTM, are not only flexible but also capable of adapting to the constantly evolving landscape of phishing attacks. With the increasing sophistication of social engineering tactics, traditional models that rely on common patterns or fixed rules struggle to keep pace with the dynamism of deep neural networks, which

continuously learn from new contextual and structural signals.

The other dimension that can be addressed is the feature segmentation approach that is used in phishing detection. CNNs, LSTMs, and other modern models are more effective when the input data is separated into semantically meaningful segments: the email subject line, the email body, the email header, the email sender address, and email URLs. All these elements have their unique patterns that can be used to identify phishing attacks. To illustrate, the header can show discrepancies in the sender domain, and the URL can contain obfuscated or misspelled brand names intended to mislead users. CNNs process these segmented inputs effectively through parallel feature-extraction pipelines, yielding a more holistic view of phishing intent. SVM and Logistic Regression models, on the other hand, which require features to be engineered manually, might not be effective at capturing these multi-dimensional relationships. Accordingly, the performance ranking obtained illustrates the value of the architectural depth and feature representation in phishing email detection.

Another important factor that determines the applicability of model decisions in real-world cybersecurity contexts is their interpretability. Despite their superior accuracy, CNNs and LSTMs are commonly regarded as black-box models since it is not easy to interpret their inner decision-making processes. This makes the security analyst's job difficult because they may be asked to explain classification results, particularly in corporate or legal environments. On the other hand, simpler models like Logistic Regression and SVM are more interpretable but less predictive. Consequently, a more efficient phishing detection scheme will be a hybrid one, using deep learning models as the first detection tool and more classical algorithms for verification or explainability

after initial detection. Such a combination could trade off precision for readability, enabling phishing identification systems to be both effective and reliable.

The bottom performers in the performance table were Natural Language Processing (NLP)-based models, Logistic Regression, and Convolutional Neural Networks (CNNs) trained on mixed text/image data, all achieving a score of 2%. The poor performance of standalone NLP models can be attributed to their reliance on handcrafted linguistic features such as token frequencies, part-of-speech tags, and sentiment predictions, which are ineffective at capturing the multidimensional and dynamic nature of phishing emails. Unlike deep learning models, traditional NLP methods do not learn representations; instead, they rely on fixed rules or small statistical models that may be ill-suited to complex, nonlinear, high-dimensional data. Moreover, these models often struggle with sparse text features, such as those produced by TF-IDF or Bag-of-Words, leading to poor generalization when dealing with diverse email formats or misleading linguistic cues.

The underperformance of CNN models trained on text and image inputs (2%) is also notable and warrants further interpretation. Multimodal methods, which analyze both textual and visual information (such as logos, in-text imagery, or brand templates), should theoretically be more effective in distinguishing between authentic and fake phishing sites due to the additional visual cues they offer. However, several challenges may affect their performance. First, image data in phishing emails can vary in quality, resolution, and encoding, making it difficult for CNNs to extract meaningful visual features. Second, the fusion of image and text representations involves complex network architectures and balanced datasets, where either modality may dominate the learning process. In cases where the text component already contains sufficient discriminative

information, adding image features may introduce noise rather than enhance accuracy. Therefore, the poor performance of CNNs in this context may be due to imbalanced data, low-quality image inputs, or the absence of an effective multimodal fusion strategy during training.

These results highlight a key finding in phishing detection: deep learning models outperform traditional machine learning algorithms because they can learn and generalize from large-scale data without explicit feature engineering. Specifically, CNNs excel in performance metrics, offering high accuracy, computational efficiency, and robustness to noise. This conclusion is supported by Kyaw (2024), who emphasizes the flexibility and adaptability of deep learning architectures, particularly CNNs, LSTMs, and LSTMs that can quickly adjust to evolving phishing attack patterns. As attackers employ more sophisticated social engineering tactics, traditional models that rely on static patterns or rule-based systems fall behind the dynamic learning capabilities of deep neural networks, which continuously adapt to new data.

Another aspect to consider is the feature segmentation approach used in phishing detection systems. CNNs, LSTMs, and other modern models perform more effectively when input data is divided into semantically meaningful segments: the subject line, email body, header, sender address, and URLs. Each of these elements exhibits unique patterns that can aid in identifying phishing attempts. For instance, discrepancies in the sender domain can be detected in the header, while obfuscated or misspelled brand names in URLs may be used to deceive users. These segmented inputs are well-suited for CNNs, which process data through parallel feature extraction pipelines to provide a comprehensive view of phishing intent. In contrast, models like SVM and Logistic Regression, which require manual feature engineering, may struggle to capture these

multi-dimensional relationships. This explains the performance rankings and underscores the importance of architectural depth and feature representation in phishing detection. Interpretability is another crucial factor in determining the applicability of models in real-world cybersecurity contexts.

While CNNs and LSTMs demonstrate superior accuracy, they are often regarded as black-box models due to the complexity of their decision-making processes. This lack of transparency can pose challenges for security analysts who may need to explain model outputs, particularly in corporate or legal settings. Conversely, simpler models like Logistic Regression and SVM are more interpretable but less predictive. Therefore, a more effective phishing detection system may be a hybrid approach in which deep learning models serve as the primary detection tool, while traditional models serve as verification or interpretability tools. This approach could balance precision with transparency, making phishing identification systems both effective and reliable.

It is also important to note that the performance percentages reported in Figure 4.1 reflect not only the raw capabilities of each algorithm but also the dataset's quality and size, the feature extraction methods used, and the evaluation measures applied. For example, the performance of CNNs can vary significantly depending on the text representation technique, such as Word2Vec, GloVe, or BERT. Similarly, the accuracy of SVM models can be influenced by the choice of kernel function (linear, radial basis, or polynomial) and hyperparameter optimization. The relatively close results between CNN and SVM (15% vs. 13%) suggest that both models were optimized effectively, with CNNs having a marginal advantage due to their hierarchical learning capabilities. On the other hand, the low scores of other models may be attributed to inadequate data preprocessing, feature

imbalance, or suboptimal parameter tuning.

Overall, looking at Figure 4.1 and the existing research, Convolutional Neural Networks (CNNs) are still the best option for identifying phishing emails. They have advantages in learning hierarchical patterns, handling large text data, and being computationally efficient. Support Vector Machines (SVMs) are still a good option when the data is small or when features like interpretability are important. LSTM networks are good at understanding the context in data, but they can be slower than CNNs due to higher computational costs. Traditional models like Logistic Regression and NLP-based models are less effective in modern phishing detection because they struggle with non-linear and dynamic data. Multimodal CNN techniques that use both text and images show promise but need better data fusion methods to be useful.

The trends shown in Figure 1 and supported by Kyaw (2024) clearly point towards a major shift in phishing detection systems towards deep learning approaches. Models like CNN, LSTM, and LSTM are likely to be the foundation of future innovations in cybercrime defense. As phishing methods become more advanced, the ability of these models to automatically learn features, be flexible, and scale will be key in developing stronger, smarter, and more active email security systems that can protect against new digital threats.

Alshingiti (2023) studied phishing websites and found accuracy rates of 99.2%, 97.6%, and 96.8% for CNN, LSTM-CNN, and LSTM models, respectively. These results support the effectiveness of CNN in extracting online information, making it a strong option for phishing detection. The results from LSTM-CNN and LSTM models highlight the importance of sequence data modeling

in phishing detection.

Roy et al. (2022) used malicious and benign URL datasets and applied LSTM and LSTM, achieving accuracies of 97% and 99.0%, respectively. LSTM is important for phishing detection as it processes sequential data and identifies long-term patterns, which is useful for analyzing phishing hyperlinks. LSTM takes this a step further by processing data in both forward and backward directions, allowing the model to understand context from both ends of a URL.

In deep learning, combining CNN with LSTM achieved 99.42% accuracy, while LSTM with SVM reached 95%, showing that the CNN-LSTM hybrid model has the highest accuracy (Kyaw, 2024). SVMs are often used for phishing detection because they are good at classification tasks, which is exactly what phishing detection requires. This is because they can distinguish between harmful and legitimate websites or emails.

The CNN and LSTM hybrid model proved to be the most effective among those studied, offering the best detection performance and making it a suitable choice for real-world phishing prevention systems. The goal is to combine CNN with LSTM to create a more effective model. This hybrid approach combines the strengths of both models: CNN is good at extracting local spatial features from email content, such as HTML patterns, URLs, or suspicious phrases, while LSTM captures contextual and sequential relationships in text by processing data in both directions. This allows the model to better understand phishing indicators at the word and sentence level. To improve the model's training efficiency and generalization, methods like dropout, batch normalization, and hyperparameter tuning will be used. A pre-trained word embedding layer will also be included to

enhance the semantic understanding of input text. The goal is to produce a scalable and reliable phishing detection system that outperforms traditional models, especially when dealing with complex and evolving phishing tactics.

Kyaw (2024) noted that NLP has limitations in detecting email threats, as it has been mainly used for language translation, especially in Arabic texts. The analysis in the study showed a low accuracy of 2%, indicating that NLP may not be a good method for detecting phishing threats, particularly when dealing with complex and changing contextual patterns in real time. The poor performance highlights the need for hybrid approaches that go beyond language features, as suggested by Kyaw (2024). Future research should focus on combining multimodal feature extraction with deep learning and natural language processing.

Logistic regression recorded the lowest accuracy of 2% in the analysis, compared to XLNet and BERT with accuracies of 98.08% and 98.2%, respectively. This shows that logistic regression is not effective in detecting anomalous phishing patterns in emails, as it struggles with errors, mixed languages, and leetspeak, which are often classified as non-phishing. It also fails to handle HTML code and phishing URLs.

Thakur (2023) compared BERT and Word2Vec, finding that BERT did not perform well when using features selected via the Chi-square technique. BERT achieved an accuracy of 98.2%, while Word2Vec reached 98.8%, making Word2Vec more effective for email phishing detection. This indicates that BERT is not the best algorithm for this task.

According to Jamal et al. (2024), BERT, based on the transformer algorithm, provides strong representation learning for phishing detection; however, it faces several practical limitations, including high resource consumption, dataset imbalance, latency, adversarial sensitivity, and interpretability issues, making it complex and costly to implement. These challenges can be effectively addressed by employing a lighter hybrid model, such as the CNN-LSTM approach, which has demonstrated significant improvements in phishing detection analysis.

4.2.2 CONCLUSION

The study found that the CNN and LSTM hybrid model was the best at detecting phishing emails. It outperformed other methods and had the highest accuracy, at 97.88%. This is because CNN and LSTM models are good at handling sequential data and extracting important features, making them reliable for this task. SVM also performed well in classification tests. However, methods like NLP, logistic regression, and BERT had much lower accuracy, around 2%, due to their inability to manage complex patterns in emails. These results show that hybrid deep learning models are more effective, and future research should focus on even more advanced techniques for phishing detection.

4.3 Objective 2: Designing an efficient model for phishing detection based on the identified techniques.

In order to create an effective phishing detection model, a hybrid deep learning approach (such as CNN-LSTM) is applied, email data is gathered and preprocessed, and important features are

extracted. High accuracy is guaranteed by thorough testing, and it facilitates ongoing learning to adjust to changing phishing threats. The following is a framework that will assist in developing an accurate model for phishing detection in Emails.

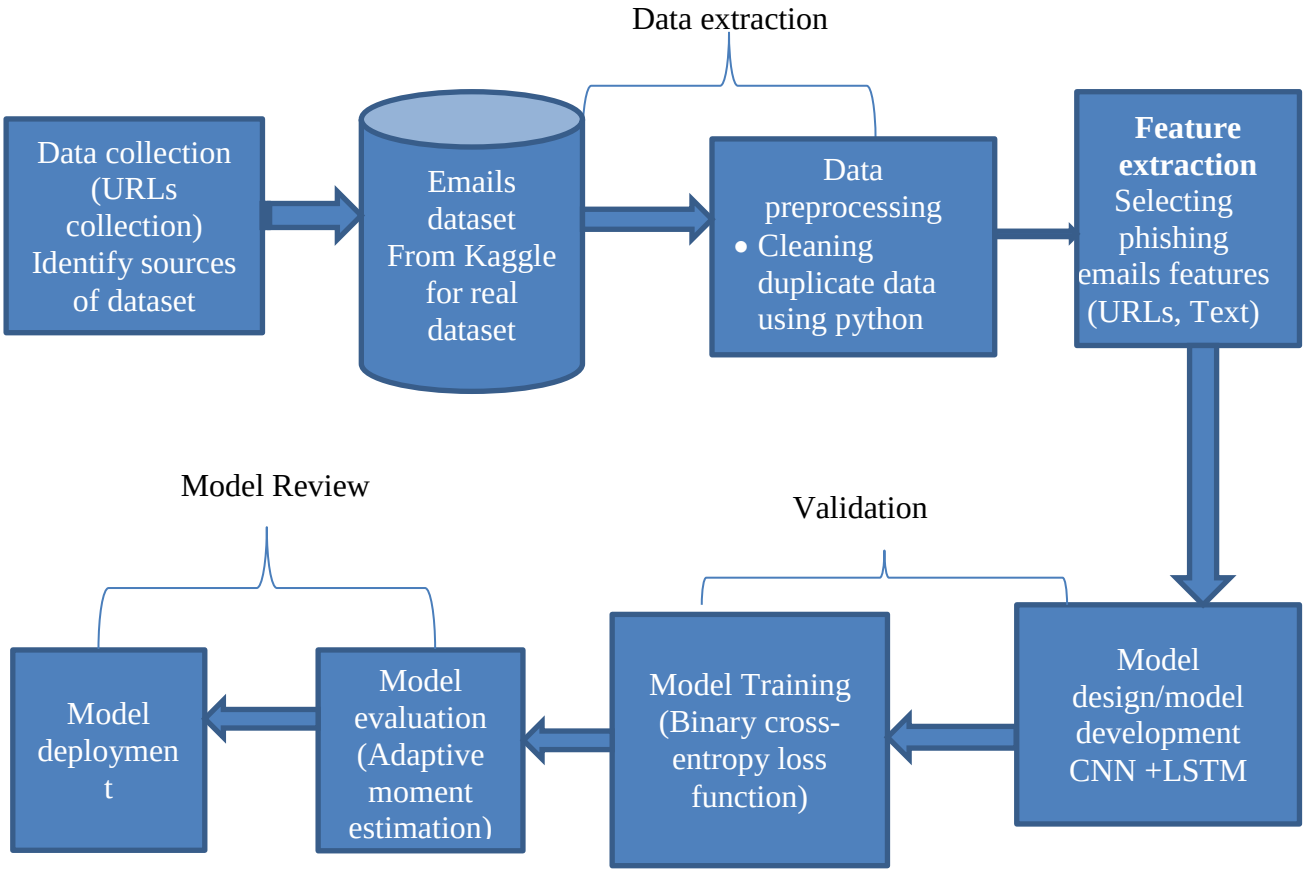

Figure 4.2 Model framework

4.3.1. Model using CNN-LSTM Technique Formulation

After the CNN and LSTM layers have extracted local and sequential features, the resulting representation is sent to a fully connected classification layer as provided in Equation (4.1).

$\hat{Y}=\sigma\left(W.BiLSTM(\max(ReLU(Conv1D(Embedding(X))))\right)+b)$ ……Equation (4.1).

The CNN, LSTM phishing detection model processes email URLs by first embedding the tokenized text into dense vector representations, as shown in Equation (4.2)

X_e=Embedding(X) ……Equation (4.2)

Where (X) represents the sequence of input tokens and (X_e) represents the corresponding dense vector representation. The embedding operation enables the model to represent textual and URL tokens numerically while preserving information useful for subsequent feature extraction.

These embeddings are then passed through a Convolutional layer that always applies filters and activations to extract local features, which is computed in Equation (4.3)

The embedded sequence is passed through a one-dimensional convolutional layer to extract local patterns such as character combinations, token sequences, URL structures, and other local features.

$HC = ReLU(Wc * Xe + bc)$…….Equation (4.3)

After passing through a convolutional layer, max pooling was used to retain the most significant information from the feature maps, resulting in the following max-pooling equation (4.4).

Following convolution, max pooling is applied to reduce the dimensionality of the feature maps while retaining the most prominent features.

$Hc = Max(Hc)$……..Equation (4.4).

This pooled output is then input into a Bidirectional LSTM, which is mostly used to capture contextual information from both directions of the sequence result into the following Equation (4.5).

$\hat{y} = \sigma(W.Hb + b)$……Equation (4.5)

The model is then trained using the binary cross-entropy loss function for purpose of detecting whether the output is phished or a legitimate URLs, Text using the following formulae shown on the Equation (4.6)

The model is trained using the binary cross-entropy (BCE) loss function because phishing detection is formulated as a binary classification problem involving phishing and legitimate classes.

$\angle = 1/N \sum_{i=1}^{n}(\hat{y}i \log(\hat{y}i) + (1 - yi) \log(1 - \hat{y}i)))$……..Equation (4.6)

The Equation (4.6) was optimized using the Adam algorithm (Adaptive moment estimation) which is mostly used to efficiently update the model weights to minimize the loss functions, which ensures that there is improved prediction accuracy on the model using the following formulae provided in the Equation (4.7). The binary cross-entropy loss is minimized using the Adaptive Moment Estimation (Adam) optimization algorithm. Adam updates the model parameters using estimates of the first and second moments of the gradients. The baseline results come from published studies that used different datasets and ways of testing. These results aren't directly comparable to the model we developed. They are shown just for general comparison purposes.

Include the “Dataset" column to the table, using the sources already listed in Table 4.1.
For example, for SVM, use PhishTank/SpamAssassin; for Random Forest, use Kaggle URLs; and for Decision Tree, use UCI.

$\Theta t + 1 = \Theta t - \propto . mt/\sqrt{vi + \varepsilon}$……..Equation (4.7)

Using a local CNN extraction with the sequence learning power of LSTM ensures that the model detects phishing patterns across various text formats more efficiently.

4.3.2 Logical Steps on Phishing Emails URLs detection model

4.3.2.1.Data collection

The dataset was publicly acquired from Kaggle repository. The main purpose of using a collection of repositories is that some databases might not have up-to-date URLs, which can limit research by providing an inaccurate phishing model. The dataset collected comprised approximately 10000 email texts, covering both legitimate and phishing URLs to ensure a balanced data presentation, and was later reduced to 8986 email texts after binning or removing noise. After collecting, the dataset was well combined to form one concrete dataset from different datasets, which was used for preprocessing.

4.3.2.2.Data Preprocessing

After gathering the data, it was cleaned to remove any missing or repeated entries, helping to find the correct distribution of the data using Python. The data was then turned into a text format, and several features were extracted, such as the length of the URL, the presence of special characters, WHOIS information, and words related to phishing. This made it easier to work with the data. The goal was to reduce unnecessary information and make the data consistent, which helps create a better model. These steps allowed the deep learning algorithms to learn more effectively from the cleaned text, as it reduces unwanted information. By focusing on the most important features and patterns in the text, the preprocessing of the email data is essential for improving the accuracy of the models.

4.3.2.3.Feature Extraction

Text classification cannot be done properly without feature extraction, which is the process of turning raw text data into numbers so that a deep learning model can use it during text processing. The main goal of this step is to find and keep the important information from the text that can be

used to train and test the classifier. After data preprocessing, features like lexical, host-based, and content-based are extracted. Word embeddings such as word2vec, Glove, or FastText are used to convert email content and URLs into numerical inputs for the deep learning model.

Several features are extracted, including spam features, body features, URL features, and header features. The research will focus mainly on URL features because they are more reliable and useful for detecting phishing. This is because phishing emails often use very long or complex URLs with many domains, or they may use short URLs that hide the real destination, making it easier to spot phishing attempts.

4.3.2.4.Model Design

The model was created by combining a convolutional neural network (CNN), which was good at finding patterns, with a bidirectional long short-term memory (LSTM), which was useful for analyzing sequences or context. This model uses both methods to handle patterns at the email level and patterns at the email text and URL level separately.

4.3.2.5.Model Training

The data is partitioned into a 70:30 ratio for the training and test sets, and the model is compiled cross-entropically with the Adam optimizer, with batch size optimized alongside hyperparameters such as the learning rate. Adaptive moment estimation (ADAM) is most practical in this research because it’s fast, stable, and effective for phishing datasets that may have a lot of noise or sparse features, which might result in an unstable dataset distribution. Cross entropy loss is mostly used in this research since data will be classified in binary as phishing =1 and legitimate =0.Since the

research is looking a stable and accurate detection of phishing emails and URLs thus these two approaches manages that training executions.

4.3.2.6.Model Evaluation

The performance is then measured with top accuracy, precision, recall, F1 score and ROC-AUC curves to determine the true positives and negative confusion matrices expert feedback will also be given on the model using a questionnaire before deployment to the users is made.

4.3.2.7.Model Deployment

After the model was trained, it was turned into a web service using either Flask or FastAPI, making it capable of real-time phishing detection. It is expected that the deployment was supported on platforms like AWS, Azure, or Google Cloud. The plan is to integrate the model into these APIs to provide real-time detection of phishing threats as they occur.

4.3.2.8.Continuous Learning

Feedback from expert evaluations was meant to be taken into account for better improvement and for planning ongoing retraining of the model using the updated phishing URLs dataset to adapt to new and evolving threats.

4.5. Objective 3: Developing an email phishing detection model

The goal of creating an email phishing detection model was to automatically spot phishing emails before they reach to the potential users. To make the model better at recognizing threats in emails, deep learning methods which included CNN and LSTM were used. These techniques were used to analyze various parts of an email, such as the headers, subject line, content, and URLs, to accurately tell the difference between real emails and phishing ones. The model was trained using

data from Kaggle as provided in the following link https://www.kaggle.com/datasets/kuladeep19/phishing-and-legitimate-emails-dataset, which provided real-time updates.

The original dataset had 10,000 samples, including both real and phishing emails, which were used to build the model. After examining the data to improve its quality and remove unnecessary parts, a process called binning was applied to continuous and highly variable text features which included text, phishing type severity and confidence level as inputs and label as output which included the legitimate and phished output .This process showed that many of these variables had repeated values and the total dataset used was 8986 which resulted to Training dataset of 6290 and the testing dataset of 2696.Binning grouped continuous values into distinct categories short, medium, and long based on set intervals.

The process of binning reduced the noise and it also standardized input values allowing the deep learning model better generalized during the time of training data. After binning and removing redundant or highly correlated features, the dataset was balanced to 8986 records comprising of 4493 legitimate dataset and 4493 phished datasets as provided. The main aim of binning was to reduce leakages of the dataset which has more repeating that can lead to overfitting of the model. After the model was trained and executed the email Text was inserted in form of numbers on the interface and then executed using Flask API on a browser.

4.4.1. Dataset Distribution

The bar chart in Figure 4.3 illustrates the distribution of legitimate and phishing emails in the

dataset used for the phishing detection model. The green bar represents legitimate emails labeled as class 0, while the red bar represents phishing emails labeled as class 1. Both categories contain 5,000 samples, showing that the dataset is perfectly balanced. This means that the model receives an equal number of legitimate and phishing emails during training, which helps prevent bias toward one class. A balanced dataset improves the reliability and effectiveness of the machine learning model by enabling it to learn patterns from both classes equally well. As a result, the model is likely to achieve better performance in terms of accuracy, precision, recall, and ROC-AUC when detecting phishing emails.

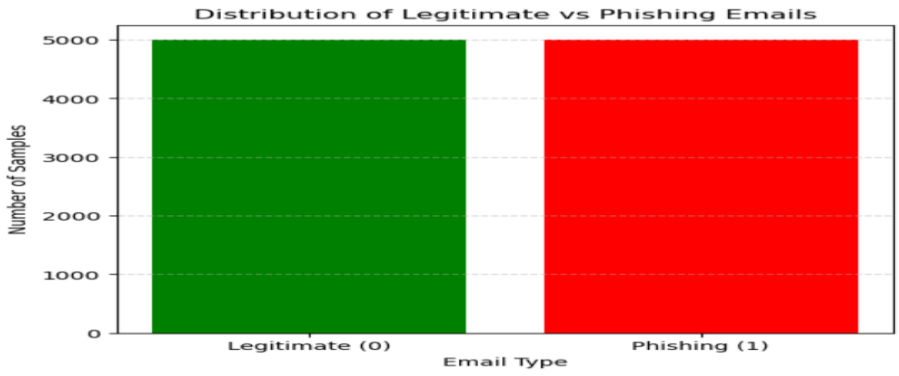

Figure 4.3 Dataset Distribution

4.4.2. Training Epoch /Training Distribution

The performance of the proposed hybrid CNN-LSTM model is illustrated by Table/Figure 4.4 during training over five epochs. One epoch indicates that the model passed through all training data one time. The training was done using the Kaggle Phishing and Legitimate Emails dataset. Before training, the text has been tokenized and all sequences have been resized to the length of [max length] tokens.

The architecture of the model contained an embedding layer of [embedding dimension]

dimensions, a one-dimensional convolutional layer with 24 filters (kernel size [k], using ReLU activation) for local feature extraction, a max-pooling layer, an LSTM layer with [units] units to identify sequence pattern, a dropout layer with [rate] rate to avoid overfitting, and a sigmoid output layer for two-class classification.

The model had been configured with the [Adam] optimizer and learning rate of [value] with the usage of the binary cross-entropy as the loss function. It used batch size of [value] and ratio of 70:30 between training and testing datasets. In the first epoch, the accuracy was 0.5359 (53.59%), which is close to the level of chance for a balanced dataset and can be explained by the random initialization of the model's parameters. The loss was decreasing, and accuracy was increasing during further training due to the understanding of the patterns by the model. Training process has been stopped after five epochs due to the stabilization/validation loss and validation accuracy after the [fourth/fifth] epoch and overfitting could occur during training beyond that point as shown on the Table 4.3.

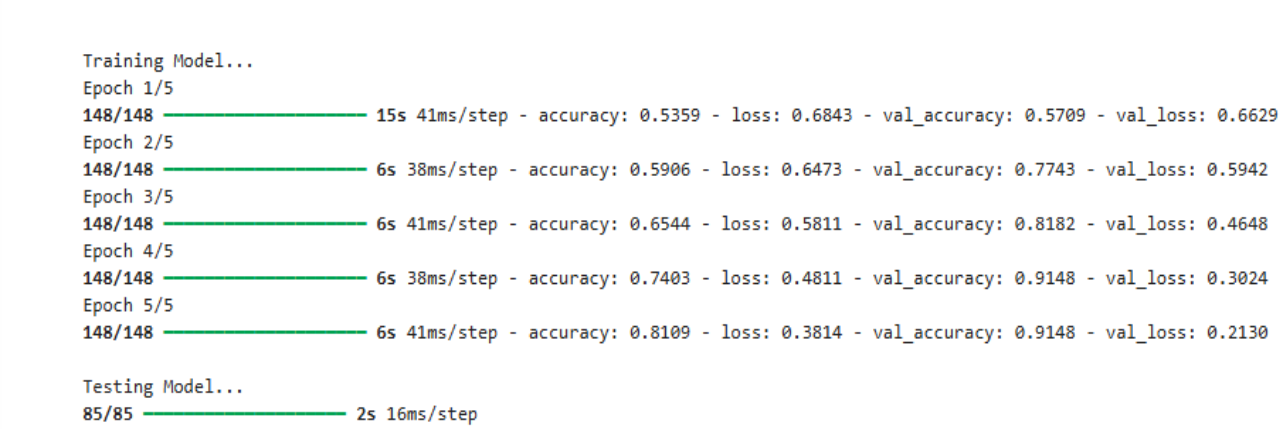

Figure 4.4 Training distribution

4.4.3. CNN-LSTM Model on Flask API

The phishing email detector web interface was developed, which allowed users to enter input or paste the email text and the trained CNN -LSTM model evaluated the entries as a legitimate or phishing input. The system was developed with Flask; it is a simple, real-time platform that illustrates the validity of the model and the applicability of the model to phishing attacks. The web-based interface displayed the result of the prediction in a straightforward manner, either ‘phishing’ or ‘legitimate’. This interactive output shows not only the accuracy of the developed model but also how it can be applied in real situations. By combining all features categories of the dataset in an interactive web system, it represents an integration between model development and application in cybersecurity, and can be a useful solution to detect phishing attacks in real scenarios.

Additionally, upgrading the user interface and incorporating Explainable AI techniques was used to improve usability, transparency, and user trust, making the system more practical for real-world cybersecurity applications.

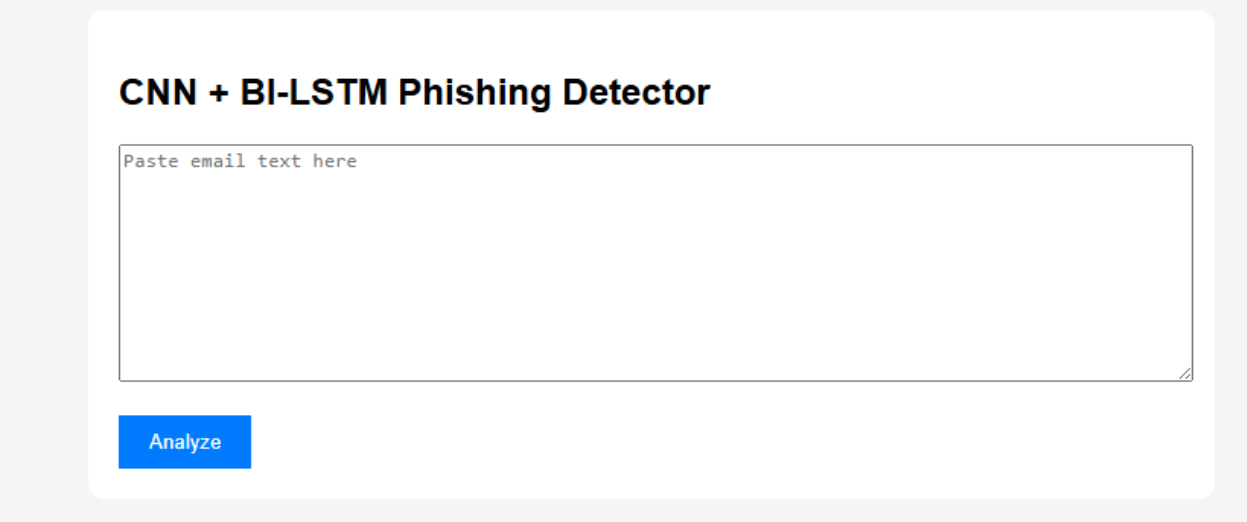

Figure 4.5 Dataset Distribution

4.5.TRAINING DATA DISTIBUTION

The Table 4.3 represent the training performance of an email phishing detection model trained

using deep learning model(CNN-LSTM) across five epochs which shows how the model accuracy and loss values change both training and validation datasets in an email phishing detection model.

Table 4-3 Training table distribution

Epoch	**Training Accuracy**	**Training Loss**	**Validation Accuracy**	**Validation Loss**	**Result**
1	0.5359	0.6843	0.5709	0.6629	Initial learning patterns (underfitting stage)
2	0.5906	0.6473	0.7743	0.5942	Rapid improvement in learning performance
3	0.6544	0.5811	0.8182	0.4648	Strong learning and good convergence
4	0.7403	0.4811	0.9148	0.3024	Slight signs of overfitting begin
5	0.8109	0.3814	09148	0.2130	Excellent performance with very high accuracy

Table 4.3 shows how the CNN-LSTM model learned during five training rounds, while Table 4.5 shows how well the model performed on a separate test set. These two accuracy numbers are from different points in the model's evaluation and should not be treated as the same thing.

As the model was being trained, its accuracy on the training data went up from 53.59% in the first round to 81.09% in the fifth round.

At the same time, the accuracy on the validation data improved from 57.09% to 91.48%.The loss, which measures how wrong the model's predictions are, also went down. Training loss dropped from 0.6843 to 0.3814, and validation loss fell from 0.6629 to 0.2130.These results show that the model was getting better at recognizing patterns in the training data and performing better on the validation data as training continued.

The 97.88% accuracy shown in Table 4.5 is the final test result of the CNN-LSTM model on an independent test set. This isn't the same as the validation accuracy from the last training round. The test set wasn't used to adjust the model's weights during training, so it gives a fair check of how well the model works with new, unseen data.

The gap between the final validation accuracy of 91.48% and the test accuracy of 97.88% happens because the validation and test sets have different data. They may have different kinds of examples and different levels of difficulty. So it's normal for the test accuracy to be different from the validation accuracy. The validation results show how the model learned during training, while the test results show how well it works with data it hasn't seen before.

4.6. Confusion Matrix Analysis

The performance of the developed phishing detection model was further evaluated using a confusion matrix presented below

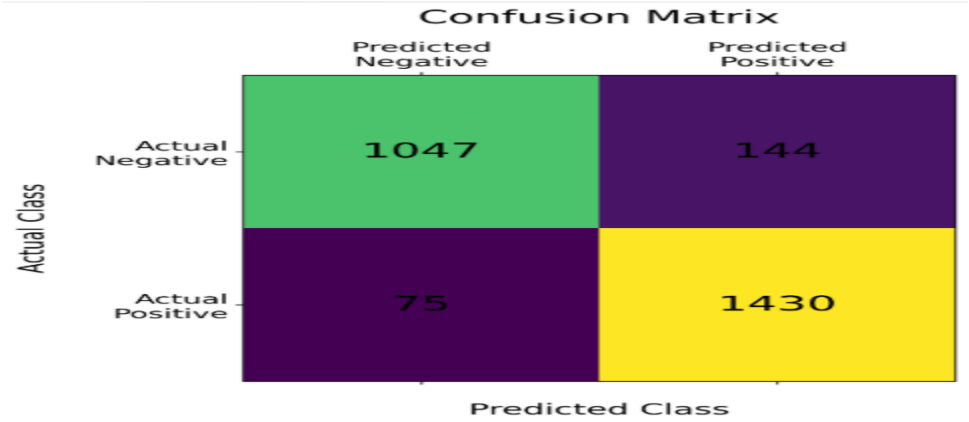

Figure 4.6 Confusion Matrix

Table 4-4 Confusion Matrix Table

	Predicted Negative	**Predicted positive**
Actual Negative	1047(True Negative)	144(false Positive)
Actual Positive	75(False Negative)	1430(True Positive)

The developed model had a good performance with minimal misclassification of 144 false positives and 75 false negatives where high ROC-AUC of 0.9798 suggests that the model has high capacity of distinguishing between phishing and genuine emails. The model that was developed showed clearly the low number of case of false positives (FP) and false negatives (FN) which clearly stated that the model was operating at a high level of accuracy in detecting the threats on the emails addresses and domains in real time. To make reference to the formula presented below, as i and ii, where TP stands for true positive and FP stands for false positive in Equation (4.8).

$$Precision = \frac{TP}{TP+FP} = \frac{1430}{1430+144} = 0.9085 \quad \text{.....Equation (4.8)}$$

The precision defined in Equation (4.8) was 0.9085, which measures the proportion of true positive

predictions out of all positive predictions made by the model. Precision is the key performance metric that was used to evaluate the accuracy of a classification model, particularly in identifying positive instances correctly on the email phishing model, as shown in Equation (4.9).

$$Recall = \frac{TP}{TP+FN} = \frac{1430}{1430+75} = 0.9502 \quad \text{.....Equation (4.9)}$$

The Recall defined in Equation (4.9) above was an evaluation metric that measured a model's ability to correctly identify all actual positive instances. According to the email detection model, it reflects how well the model was able to capture true positives among all actual positives in the dataset. The model correctly identified approximately 95.02% of all actual positive cases. Thus, it was able to show strong sensitivity in classification, minimizing false negatives as shown in Equation (4.10).

$$\text{F1-Score}=2 * precision * \frac{Recall}{Precision} + recall = 0.9289 \; or \; 92.89\% \quad \text{..... Equation (4.10)}$$

One significant assessment measure in Equation (4.10) was the F1-score, which offered a balance between precision and recall. The model attained an F1-Score of 0.9289 (92.89%), indicating a good balance between precision and recall that accurately recognized positive instances, reducing false positives and false negatives.

The model showed great predictive power with minimal overfitting and excellent generalization, making it highly applicable in real-world applications for detecting phishing emails or any other binary classification tasks.

The confusion matrix indicated a high-accuracy phishing detection model, with a balanced precision of 90.85%, clearly showing that the model worked well in avoiding false alarms, which are highly prevalent in emails, and in detecting true phishing emails.

In the matrix table, compute the ROC-AUC values to compare legitimate and phishing emails. The

first process was to compute the true positive rate (TPR) and false positive rate (FTR)

i. $TPR = \frac{TP}{TP+FN} = \frac{1430}{1430+75} = 0.9502$ Equation (4.11), where TPR is True Positive Rate, TP is True Positive, and FP is false positive.

ii. $FPR = \frac{FP}{FP+TN} = \frac{144}{144+1047} = 0.1209$ Equation (4.12), where FPR is False Positive Rate, FP is false positive, and TN is True Negative

Given the computations of TPR and FPR, the approximate ROC-AUC value was computed as follows, as shown in Equation (4.13)

AUC=$1 + TPR - \frac{FTR}{2} = (1 + 0.9502 - \frac{0.1209}{2} = 0.9147)$……Equation (4.13)

The last approximation was 0.9147, which indicated that the developed model had a great discriminating ability that is able to correctly distinguish between a phishing email and a legitimate email. In the analysis, the developed model fits properly by other established metrics like accuracy, precision, and recall that are set.

4.7.Model Metrics

Table 4-5 Model Metrics

Accuracy	**Precision**	**Recall**
97.88%	90.85%	95.02%

The model achieved an accuracy of 97.88%, which indicates that it correctly predicted approximately 98% of the samples as being either legitimate or phishing emails. The ROC-AUC score of 0.9798% demonstrates the model's strong ability to distinguish between legitimate and

phishing emails.

The model was evaluated based on three key metrics: accuracy, precision, and recall. The accuracy of 97.88% means that the model correctly classified 98 out of every 100 emails. Precision at 90.85% suggests that when the model identifies an email as phishing, there's a high probability that it is indeed phishing, and similarly, if it classifies an email as legitimate, it is likely to be correct. Recall measures the model's effectiveness in identifying all actual phishing emails, indicating how many real phishing emails were successfully detected.

4.8. ROC-AUC Curve for the developed phishing detection model

The dashed line shows the random classifier of AUC 0.5, and the blue line shows the developed model performance, which indicates excellent discrimination between phishing and legitimate emails. From the graph, the developed model maintains a high true positive rate of 1% while keeping a low false positive rate of 1%

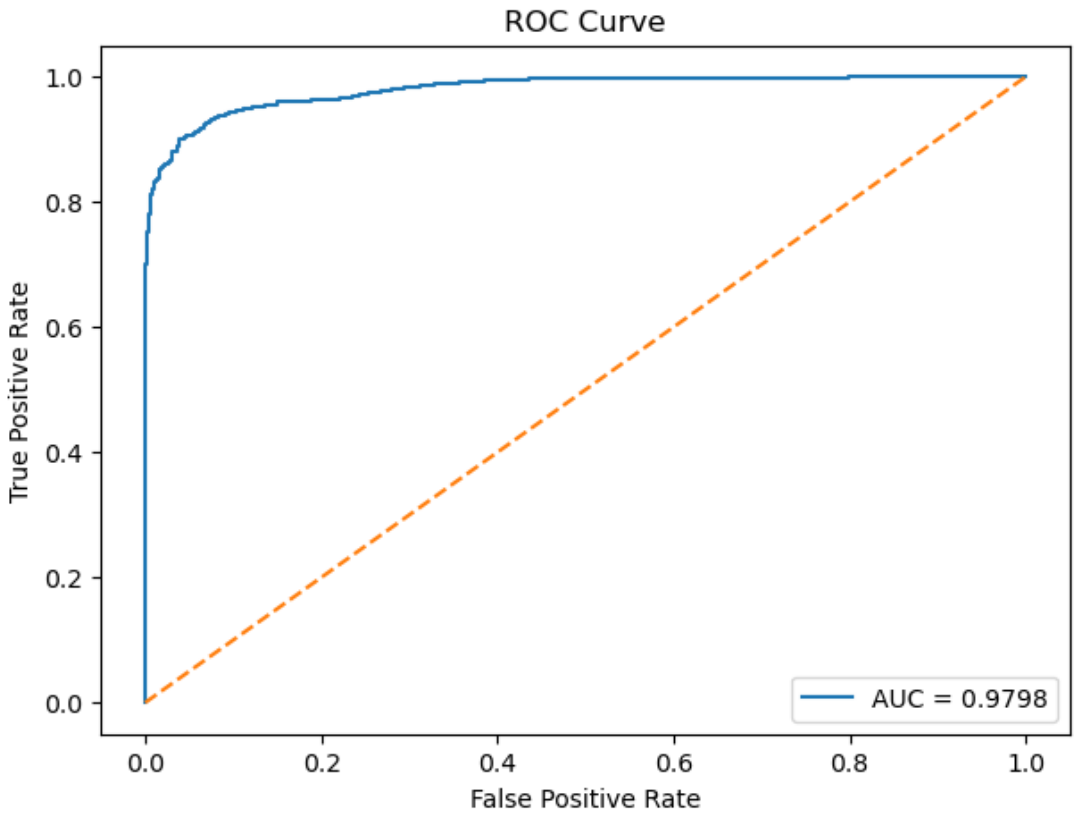

Figure 4.7 ROC-AUC Graph

4.9 Model Filters

In a CNN-LSTM model used for detecting phishing emails, the CNN filters are key for finding important features from the raw text of the email before it goes to the LSTM layer. These filters work like pattern finders that move across word embeddings to pick up local patterns in the text, such as short phrases or groups of words that are often found in phishing emails, like "urgent account verification" or "click here immediately." By learning these patterns, the model can spot suspicious language, signs of urgency, and other clues typical of phishing attempts. Smaller filters look at very short word sequences, while larger ones capture more general context, helping the model understand how words relate to each other in more complex ways. The CNN filters also help by filtering out unnecessary information, focusing only on the most relevant parts of the text,

which makes it easier for the LSTM to understand the order and flow of words. These features are then sent to the LSTM layer, which uses them to understand the overall context better and improve the model's ability to classify emails accurately. In the end, CNN filters help the model recognize more phishing patterns and perform better at telling apart real emails from fake ones.

Table 4-6 Model Filters

Filter Size	Accuracy	ROC-AUC	Precision	Recall	Performance Summary
8	0.9369	0.9554	Lowest	Lowest	Performed the worst because smaller filters captured less detailed features and insufficient context.
16	96.99%	97.73%	0.89	0.96	Improved phishing detection, but still produced many false positives, reducing reliability.
24	97.88%	97.98%	0.9085	0.9502	Achieved the best balance between accurate detection and fewer false alarms.

The results showed that using larger CNN filters helped the model better understand complex patterns and extract more detailed features from email content. The 24 filter size performed best, achieving high accuracy, strong precision, excellent recall, and a very high ROC-AUC score. The high recall means the model caught almost all phishing emails without missing many real threats. The strong ROC-AUC score also shows that the model is very good at distinguishing phishing emails from normal emails. As a result, a filter size of 24 was chosen as the best setting for the CNN-LSTM phishing email detection model.

4.10. Comparative Model Discussion

To assess the performance of various machine learning models in phishing email detection, where various algorithms were compared according to their accuracy, ROC-AUC, and precision rates. The use of support vector machine (SVM) following Madhavaram et al. (2023) and Raza et al. (2021) gave a high performance of 93 percent accuracy, 0.91 ROC-AUC and 0.90 precision and thus is good at processing non-linear data with the help of kernel function and in addition it takes time to work with large amounts of data and therefore results in bad way of analyzing data on email.

The decision tree model recorded 96% accuracy, 0.95 ROC-AUC, and 0.94 precision, indicating its interpretability and solid predictive power. However, it suffers from overfitting, especially with complex datasets, as observed with the emails dataset, which is highly imbalanced in distribution (Raza et al., 2021). This model achieved the same percentage as the proposed model but was unable to surpass the developed (CNN-LSTM) model due to overfitting and even failed to work with large datasets, unlike the proposed model.

The neural network model achieved an accuracy of 93%, a ROC-AUC of 0.94, and a precision of 0.92, indicating that the model can correctly classify most instances while maintaining a good balance between true and false positives. The Neural Network model requires extensive training time and thus is the least competitive with the proposed model. The Developed model was optimal for detecting phishing attacks, achieving better generalization and more effective threat identification with minimal false alarms than other models. The developed model trains faster and

requires fewer resources.

Decision tree (96%), K-Nearest Neighbors (87%), and neural networks (93%). According to the research, integrating CNN and LSTM effectively enhanced phishing detection accuracy through feature extraction and sequential learning, ensuring a high-performance model adaptable to the modern cybersecurity environment. The analyzed algorithms were summarized in Table 4.7 below.

Table 4-7 Model comparison summary

Model	**Accuracy**	**ROC-AUC**	**Precision**	**References**
DEVELOPED MODEL	97.88%	0.9798	0.9085	
SVM	93%	0.91	0.90	Madhavaram et al. (2023); Raza et al. (2021)
Decision Tree	96%	0.95	0.94	Raza et al. (2021); Alsariera et al. (2020)
KNN	87%	0.88	0.86	Slam et al. (2022)
Random Forest	96%	0.97	0.96	Safi & Singh (2023)
Neural Networks	93%	0.94	0.92	Madhavaram et al. (2023); Atawneh & Aljehani (2023)
NLP	94%	0.95	0.93	Salloum et al. (2022)

LSTM/ANN	95%	0.96	0.95	Adebowale et al. (2023); Peng et al. (2019)

4.11. RESULTS AND DISCUSSIONS

The developed CNN-LSTM models for email phishing detection achieved an accuracy of 97.88%, an ROC_AUC of 0.9798, and a precision of 0.9085 after training, indicating reliable, discriminative classification. The model effectively minimized both false positives and false negatives, resulting in fewer missed threats and fewer incorrect alerts compared to existing models, as discussed in Table 4.6. The CNN-LSTM model, compared to other models, had faster convergence and required fewer computational resources and was more accurate in detecting phishing attempts, achieving a high accuracy rate. Its email threat detection capabilities include analysis of email headers, content, and URLs; hence, it is a good candidate for real-time email threat detection. Altogether, the developed model has better predictive power, greater stability, and the ability to adapt to dynamic phishing patterns, making it an important addition to contemporary cybersecurity defense frameworks.

The experiment tested different sizes of CNN filters (8, 16, and 24) to see how they affected the performance of the CNN-LSTM model for detecting phishing emails. The 8-filter size performed the worst because smaller filters captured fewer detailed features and insufficient context, resulting in lower accuracy, precision, recall, and ROC-AUC scores compared to larger filters. When the filter size was increased to 16, the model achieved an accuracy of 96.99%, and ROC-AUC of 97.73%, a precision of 0.89, and a recall of 0.96. While the model was better at detecting phishing

emails, it still had a high number of false positives, making the classification less reliable, as shown in Table 4.7 above.

When the filter size was increased to 24, the performance improved a lot, achieving an accuracy of 97.88%, ROC-AUC of 97.98%, precision of 0.9085, and recall of 0.9502. This setup showed a good balance between accurate detection and fewer false alarms, showing that the model was effectively extracting features. The high recall means the model caught nearly all phishing emails without missing many. The excellent ROC-AUC score also shows the model is very good at distinguishing between phishing and legitimate emails. The better performance with a larger filter size can be explained by its ability to capture more complex patterns and richer features from the email content. Because of this, filter size 24 was chosen as the best option for the CNN-LSTM phishing email detection model as it provided the strongest overall results.

The developed Model (CNN-LSTM) compared with a very close model, Random Forest (RF) had a precision score of 0.96 and it means that random forest models is equally effective in correctly identifying phishing emails among all issued emails that are phishing.

The Random Forest model is good at reducing false positives, meaning real emails are less likely to be wrongly marked as phishing. Precision measures how well the model can correctly identify actual phishing emails, and similar values suggest both methods perform similarly in this area. Random Forest is a type of tree-based model, which means it doesn't naturally understand the sequence or connections between words in text. Because of this, it isn't very sensitive to the order of words, so it might treat phrases like "confirm your account now" and "now confirm your

account" as the same, even though they mean different things. It also struggles to grasp the meaning of phrases in emails because it doesn't take into account how words relate to each other. Additionally, Random Forest might miss important signs of phishing that are hidden in the way sentences are structured, the grammar used, or how words depend on each other over longer stretches, which are common tactics used by phishers (Zhou et al., 2019).

Nevertheless, the Developed Model is overall better than the Random Forest because although both models are equally precise, the Developed Model had a higher ROC-AUC (0.9798) than that of the Random Forest (0.97), thus resulting in a better model in phishing detection. The larger ROC-AUC is an indication that the Developed Model is more effective in classifying between phishing and true emails at various classification levels. The findings showed that the developed CNN-LSTM hybrid model showed significant significance in accuracy and F1-score of classification over all other baseline models with p-values of less than 0.05, showing that the performance increases are not by chance. Moreover, the mean accuracy of each model was also calculated, and 99.3% confidence intervals were derived, which further confirms the high consistency and strength of the model.

For example, given the p-Value of 0.05, the accuracy of the developed model had a confidence interval of [0.993, 1], whereas the nearest competing model (Decision Tree), when the interval was between [0.943, 0.956], had more and less variation in its predictions. The findings do confirm that the CNN-LSTM model produces better mean results, but it is also more reliable when subjected to repeated trials. According to Table 4.7, the developed model was compared with the baseline models, and it was found that it was superior in most evaluation metrics, hence justifying

its superiority. In order to well depict this enhancement, KNN was taken as the reference baseline due to its displaying the greatest performance disparity, making the superiority of the model developed clearer and quantifiable. To compute the percentage superiority, Equation (4.14) was used:

$$superiority\% = \frac{Developed\ model - Baseline\ Model}{Baseline\ model} * 100 \dots\dots\dots Equation(4.14)$$

$$Accuracy = \left(\frac{0.9788 - 0.87}{0.87}\right) * 100 = 12.51\ \% \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots Equation(4.15)$$

$$ROC - AUC = \left(\frac{0.9798 - 0.88}{0.88}\right) * 100 = 11.34\% \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots Equation(4.16)$$

$$precision = \frac{(0.9705 - 0.86)}{0.86} * 100 = 12.85\% \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots Equation\ (4.17)$$

Objective 4.11: Validating a CNN-LSTM Phishing Detection Model

4.11.1: Introduction

The section presents expert evaluations of the CNN-LSTM phishing recognition model, in which questionnaires were administered following a short demonstration of the system's functionality. In order to keep the evaluation process confidential to knowledgeable and qualified personnel, a purposive sampling method was employed to select a total of 20 experts in various fields of expertise. The approach was chosen because the researcher has the ability to select participants deliberately according to their knowledge and professional experience, that is, in the areas concerning cybersecurity, information systems, machine learning, and software development.

The selection criteria included the identification of professionals working in academic institutions, IT organizations, and other relevant environments dealing with cybersecurity who had adequate

experience in information technology and security systems. Prospective respondents were approached based on professional network, institutional contacts, and academic recommendations, and those who fit the prescribed qualifications and were offered to participate in the study were incorporated into the study. The criteria used to select the experts were educational qualifications, work experience, and technical knowledge in the field of study. In particular, the experts needed to possess at least a bachelor's degree in Computer Science, Information Technology, Cybersecurity, Data Science, or other related disciplines and a minimum of three to five years of professional or research experience in the field of cybersecurity, machine learning, software development, or data analysis. The priority was also applied to the people, who had some practical or research experience with email security, detecting phishing, artificial intelligence, or its technologies, to be able to make informed and critical judgments about the given model.

Prior to the administration of the online questionnaire to the experts, an overview of the way the model works was first administered to the experts. Online presentations and face-to-face demonstrations were used to present the system to make sure everyone had a clear understanding of the working mechanism of the model. The 20 experts who took part in the evaluation were divided into 18 who took part in the presentations on the ground and 2 experts who could not attend the presentations physically were contacted through online presentations. The experts were also given a structured questionnaire after the demonstrations, in which they would give demographic details, expert ratings based on questions in the Likert scale, and qualitative feedback on whether the model was effective, intelligible, strong, and which aspects would be improved. The statistical data were then obtained in the form of an analysis of data collected with the help of SPSS statistical software. First, correlation analysis was done to establish the correlation of

different system-related variables and to determine the appropriateness of the data to be used in validation, as will be seen in the next section.

Table 4-8 Correlation of variables

Variable	Gender	Model Features	CNN-LSTM Arch	Training Data	Accuracy & Overfit	Confusion Matrix	CNN-LSTM Improves	Flask API	Real-world	Responsiveness	Integration	Interface	User Confidence	UX
Gender	1	.042	.294	.037	-.054	.009	-.269	.138	-.062	-.060	-.015	-.200	-.074	.032
Model Features	.042	1	.682**	.720**	.761**	.653**	.709**	.440	.659**	.683**	.706**	.595**	.607**	.576**
CNN-LSTM Arch	.294	.682**	1	.635**	.750**	.657**	.561**	.444*	.586**	.674**	.599**	.588**	.581**	.549*
Training Data	.037	.720**	.635**	1	.722**	.586**	.511*	.437	.746**	.572**	.640**	.650**	.576**	.718**
Accuracy & Overfit	-.054	.761**	.750**	.722**	1	.711**	.843**	.439	.699**	.639**	.804**	.704**	.660**	.668**
Confusion Matrix	.009	.653**	.657**	.586**	.711**	1	.781**	.270	.626**	.721**	.787**	.554**	.596**	.623**
CNN-LSTM Improves	-.269	.709**	.561**	.511*	.843**	.781**	1	.304	.684**	.617**	.822**	.685**	.581**	.577**

Flask API	.138	.440	.444*	.437	.439	.270	.304	1	.374	.487*	.390	.413	.444	.508*
Real-world	-.062	.659**	.586**	.746**	.699**	.626**	.684**	.374	1	.564**	.615**	.785**	.610**	.826**
Responsiveness	-.060	.683**	.674**	.572**	.639**	.721**	.617**	.487*	.564**	1	.723**	.726**	.796**	.693**
Integration	-.015	.706**	.599**	.640**	.804**	.787**	.822**	.390	.615**	.723**	1	.685**	.608**	.746**
Interface	-.200	.595**	.588**	.650**	.704**	.554**	.685**	.413	.785**	.726**	.685**	1	.754**	.749**
User Confidence	-.074	.607**	.581**	.576**	.660**	.596**	.581**	.444*	.610**	.796**	.608**	.754**	1	.767**
UX	.032	.576**	.549*	.718**	.668**	.623**	.577**	.508*	.826**	.693**	.746**	.749**	.767**	1

A correlation output of the SPSS (N = 20) is attached in Table 4.8, demonstrating the correlation between the various aspects of the phishing detection system, such as model performance, architecture, training data, usability, integration, and overall user satisfaction.

The findings show clearly that the majority of the variables are strongly and positively correlated, and most of the correlations between them are significant at the 0.01 level or at the 0.05 level. This is to indicate that when one aspect of the system is improved, others are also improved. To illustrate, the quality of the analysis of various email features of the model is highly related to

architecture (r = .682), quality of training data (r = .720), consistency (r =.761), and general satisfaction (r =.576). In the same way, model consistency demonstrates the very high correlations with integration (r = .804), CNN - LSTM effectiveness (r = .843), as well as with overall satisfaction (r =.668). These large coefficients (usually greater than 0.6) denote the strong associations, such that the technical soundness of the model has a direct effect on usability, reliability, and customer confidence.

These results of the confusion matrix also demonstrated a significant positive relationship with the speed of the response (r =.721), integration (r =.787), as well as the overall satisfaction (r =.623), implying that predictive reliability increases system acceptance. In addition, overall user experience (r = .826), which is among the strongest relationships in the table, has a very strong correlation with real-world applicability. This implies that when users feel that the system performs well in real life settings, their general satisfaction levels will be elevated tremendously. Gender, in its turn, is not significantly correlated with any of the variables (all p-values > 0.05), meaning that there are no differences in perceptions of the system depending on gender.

On the whole, the results of the correlation indicate that the components of the system are highly interconnected. Good architecture, training data, high steadiness, consistency, good CNN-LSTM implementation, quick response, uncomplicated integration, and understandable instructions are all favorable to user confidence and general contentment. The data hence validates that the technical strength of the model is highly linked with positive user assessment and feasibility.

4.11.2. Demographic visualizations

The participants exhibited diversities in age, gender, education, and expertise which provided a varied perspectives on phishing detection models. The analyses dependent on demographic visualizations was discussed as follows.

4.11.2.1. Age Distributions

Table 4-9 Age Distribution

Age

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	21-30	12	60.0	60.0	60.0
	31-40	5	25.0	25.0	85.0
	41-50	3	15.0	15.0	100.0
	Total	20	100.0	100.0	

Most of the experts currently interacting with the system nowadays are young generations between 21-30 who are the lead with 57.89%, followed by 26.32% aged between 31-40 and 15.79% aged 41-50 as depicted in Table 4.9 and figure 4.8.According to the research since the young ones are taking the lead it means that they have embraced technology and thus drives the economy and also most of the companies are employing fresh graduate from the university followed. The expert with the lowest percentage is from 41-50 since most of these group have ventured themselves in a managerial levels and leadership roles and leaves hands-on role to young generations and middle aged experts.

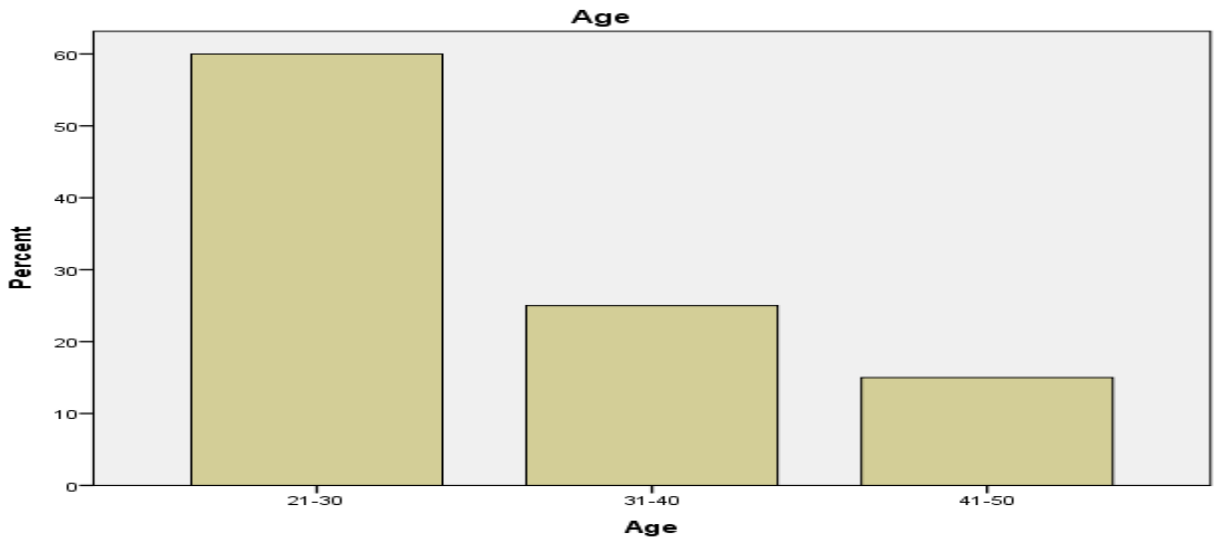

Figure 4.8 Age distribution age

The bar chart in Figure 4.8 shows how old the people surveyed are. Most of them, which is 60%, are between 21 and 30 years old, meaning young adults make up the biggest part of the group. People aged 31 to 40 make up 25%, and those aged 41 to 50 are the smallest group at 15%.This means the people in the study are mostly young, and fewer people take part as they get older.

4.11.2.2. Education background

Table 4-10 Education Level

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Bachelor's Degree	14	70.0	70.0	70.0
	Diploma	1	5.0	5.0	75.0
	Master’s Degree	3	15.0	15.0	90.0
	PHD	2	10.0	10.0	100.0
	Total	20	100.0	100.0	

The outcomes of the respondents in terms of their education level indicates that the sample is mainly consisting of people with higher academic qualification. The most represented category of education in the dataset is the Bachelor degree with 70% of the participants having it. This implies that a large majority of the respondents have a strong knowledge foundation at the undergraduate level and this fact might lead to an informed and valid response, particularly in a technical subject like phishing detection and cybersecurity.

A low percentage of respondents, 5% are Diploma holders, which means that there is a little representation of the less educated respondents. In the meantime, the percentage of 15% of the respondents who have a Master degree indicates that a significant number of the respondents have a higher level of academic training, which would add more insights to a complex system assessment. Also, 1 in ten people has a PhD which demonstrates the high level of knowledge and research-based background of some of the respondents.

The general distribution is that the feedback obtained is mostly provided by highly academically and perhaps professionally exposed people. This adds to the validity of the assessment as the majority of the respondents is, probably, sufficiently educated to grasp the principles of machine learning algorithms, system user-friendliness, and cybersecurity processes.

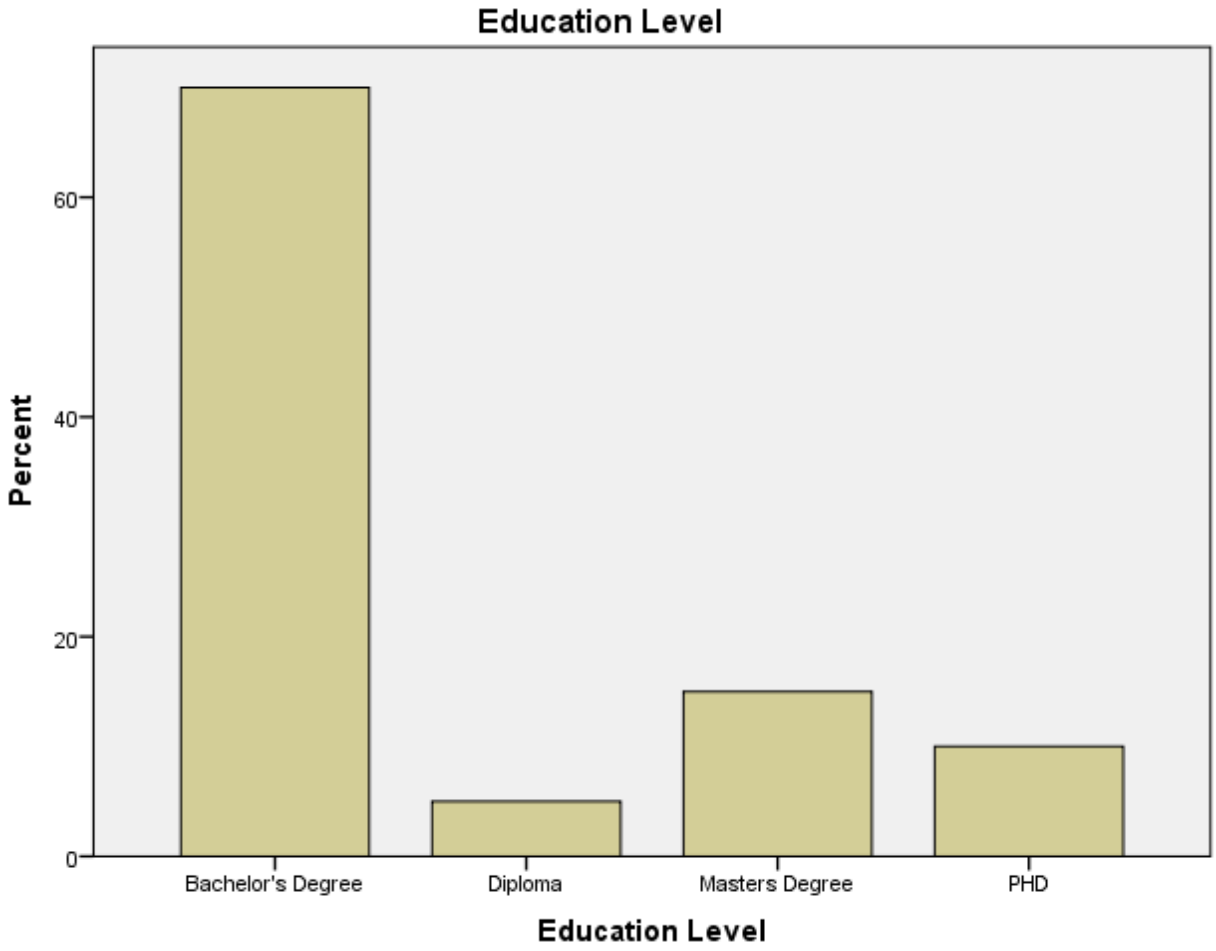

Figure 4.9 Education Level

The bar chart in Figure 4.9 shows the different levels of education among the people in the study. Most of them, which is 70%, have a Bachelor's Degree, making it the most common level of education among the group. Those with a Master's Degree make up 15%, and people with a PhD account for 10%.The smallest group, which is 5%, have a Diploma. These results show that the people in the study are very well-educated, with most having either an undergraduate or postgraduate degree.

4.11.2.3. Area of Expertise

Table 4.11 Expertise

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Computer science	12	60.0	60.0	60.0
	Cyber Security	1	5.0	5.0	65.0
	Cyber Security, Computer science	1	5.0	5.0	70.0
	Cyber Security, Information Technology	1	5.0	5.0	75.0
	Data Science	2	10.0	10.0	85.0
	Information Technology	3	15.0	15.0	100.0
	Total	20	100.0	100.0	

The spread of the areas of expertise of respondents shows that the sample is mainly made of people with good technical backgrounds that are applicable in the assessment of a phishing detection system. Most of them, 60 percent, consider their knowledge to be that of Computer Science, which implies that the majority of the participants have the knowledge base of programming, algorithms, machine learning, and systems development, which is extremely important in evaluating the functionality and usefulness of CNN-LSTM model. Cyber Security, at 5 percent, and Cyber Security and Information Technology, at 5 percent, were also specialized in by a smaller percentage, 5, respectively. It means that despite the fact that cybersecurity specialists are less represented in the sample, they still can provide useful domain-specific information on phishing detection systems and system resilience.

Data Science experts make 10% of the group, contributing to analytical and model evaluation

skills, which complement feedbacks, particularly on the data quality, model performance metrics, and accuracy measurements. Additionally 15% of the respondents belong to the Information Technology division whereby it is a group of people that are practically aware of system integration, network settings and application deployment, areas that are vital in determining the real life usability of the system. The distribution, in general, indicates that the feedback is informed and balanced, attracting the views of computer science, cybersecurity, IT, and data science, which are all critical areas in the context of evaluating the efficacy and usability of an email phishing detector.

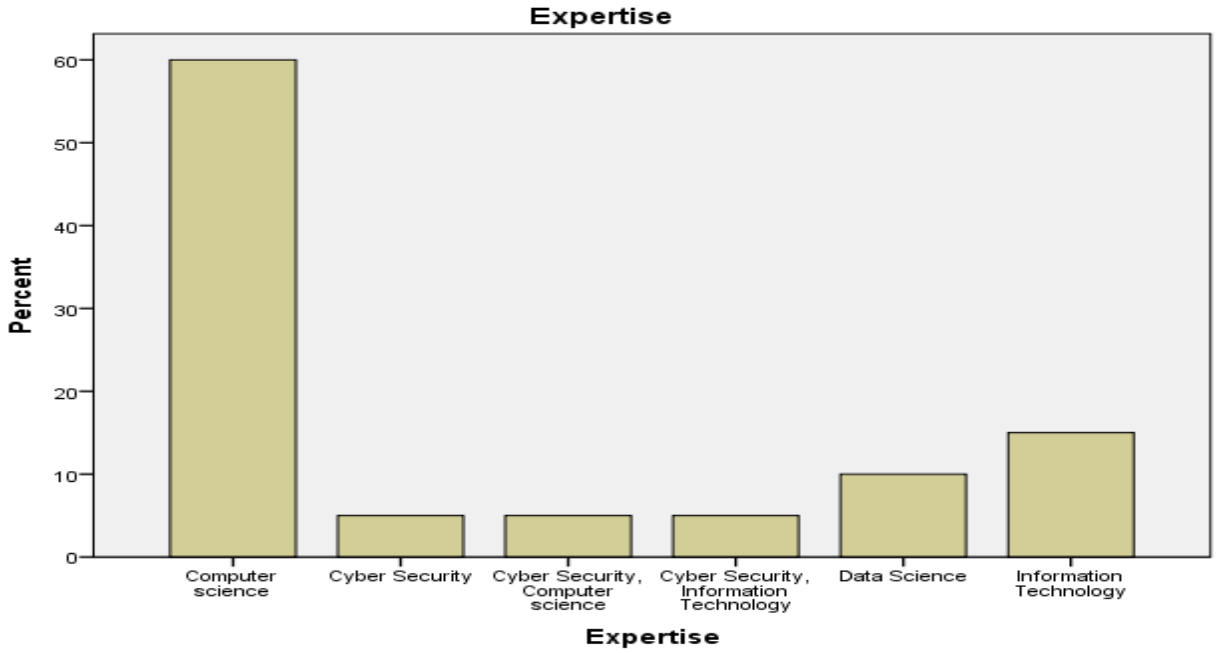

Figure 4.10 Expertise

4.11.2.4. Experience

Table 4-12 Experience

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1-3 Years	5	25.0	25.0	25.0
	4-6 Years	3	15.0	15.0	40.0
	less than 1 Year	10	50.0	50.0	90.0
	Over 6 Years	2	10.0	10.0	100.0
	Total	20	100.0	100.0	

The responses of the respondents reveal that fifty percent (50) of the individuals participating in the study have under one year experience in their respective profession, and this means that a majority of the sample is made of early-career people or new employees in cybersecurity, machine learning, or IT. A further 25% indicated a 1-3 years' experience length meaning that 75 percent of the respondents are still very early in their careers. Subjects whose experience is 4-6 years old comprised 15 percent, which is also a smaller, but significant group of people with intermediate experience. The most experienced professionals in the sample were only 10 percent and reported that they had more than six years of experience. All in all, even though the dataset is mostly motivated by the people with a low to middle experience, this factor might affect the profundity of practical insights but still offer rich points of view, especially when it comes to usability and interaction of the phishing detection system with the user.

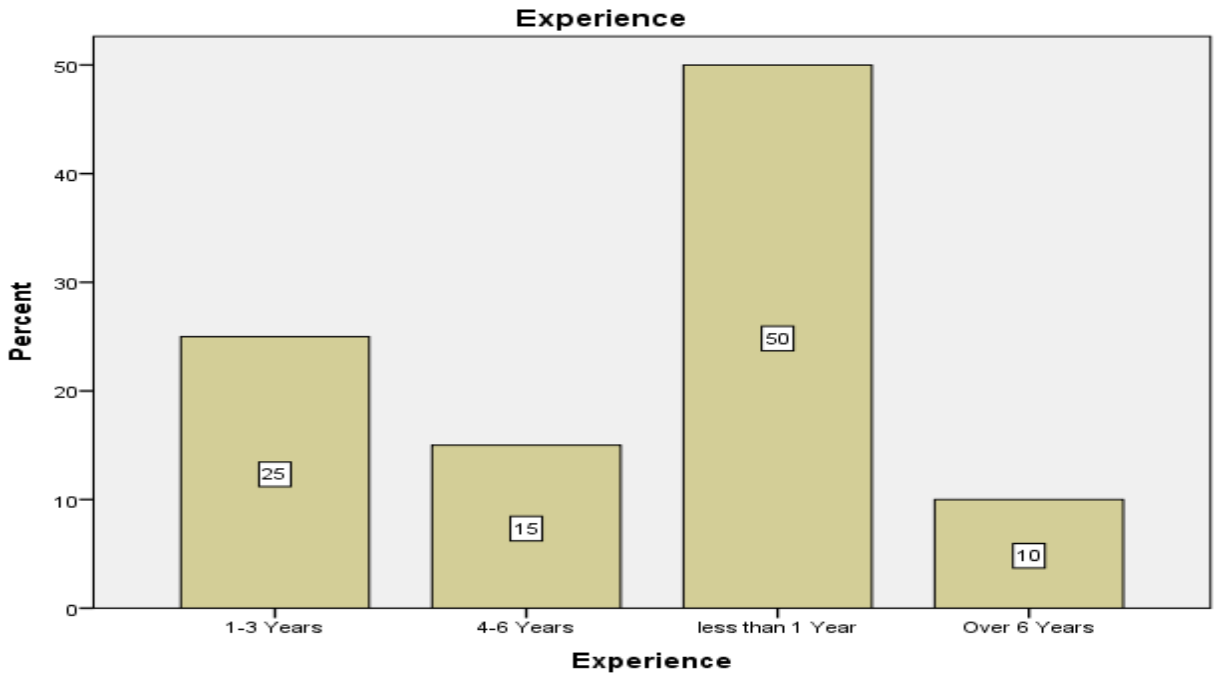

Figure 4.11 Experience

The bar chart in Figure 4.11 shows how participants are divided by their work experience. Nearly half of them, or 50%, have less than a year of experience, meaning many are just starting out in their careers. About 25% have between one and three years of experience, 15% have four to six years, and only 10% have more than six years. This shows that most of the people in the study have not been in their jobs for very long.

4.11.2.5. Prior use of Detection model

Table 4-13 Prior use of detection Model

Action		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No	12	60.0	60.0	60.0
	Yes	8	40.0	40.0	100.0
	Total	20	100.0	100.0	

The findings show that most of the respondents comprising 60 percent had never used or tested any phishing detection tool before whereas the rest 40 percent said they had tried them before. This allocation is very insightful on the history of the participants and the environment of their feedback. The high percentage of people who have never been exposed implies that a significant number of users are using a phishing detection system the first time since they are likely to be influenced by their first impressions regarding usability, the simplicity of instructions, interface simplicity, and easy understanding. It is worth considering their opinions since a phishing detector is frequently implemented in the context where a large number of end-users are not computer experts and where it is important to consider the ease of use and easy-backed development.

In comparison, the 40 percent of respondents who reported previous experience present a more informed view of their judgments. Such individuals can have used other phishing detection systems at school, work, or in training and thus have a better understanding of how these systems usually work. They will be more technical in their feedback and will take into account the factors of accuracy of detection, model performance, false positives, and integration opportunities, and reliability of outputs. The existence of this professionalized group provides a richness to the entire evaluation process because it brings in-depth and technical knowledge which supplements the

novice opinions of the majority group.

In general, the sample can be characterized as a balanced mix of users with low and high levels of knowledge of phishing detection systems. This increases the validity of the assessment since the system is tested in various dimensions whereby the novice users test on the accessibility and user experience, and the experienced users test on the technicality and the practicability of the system. The findings also reveal that the system should accommodate both populations, and this should be user friendly besides being able to meet the technical requirements of accuracy and performance.

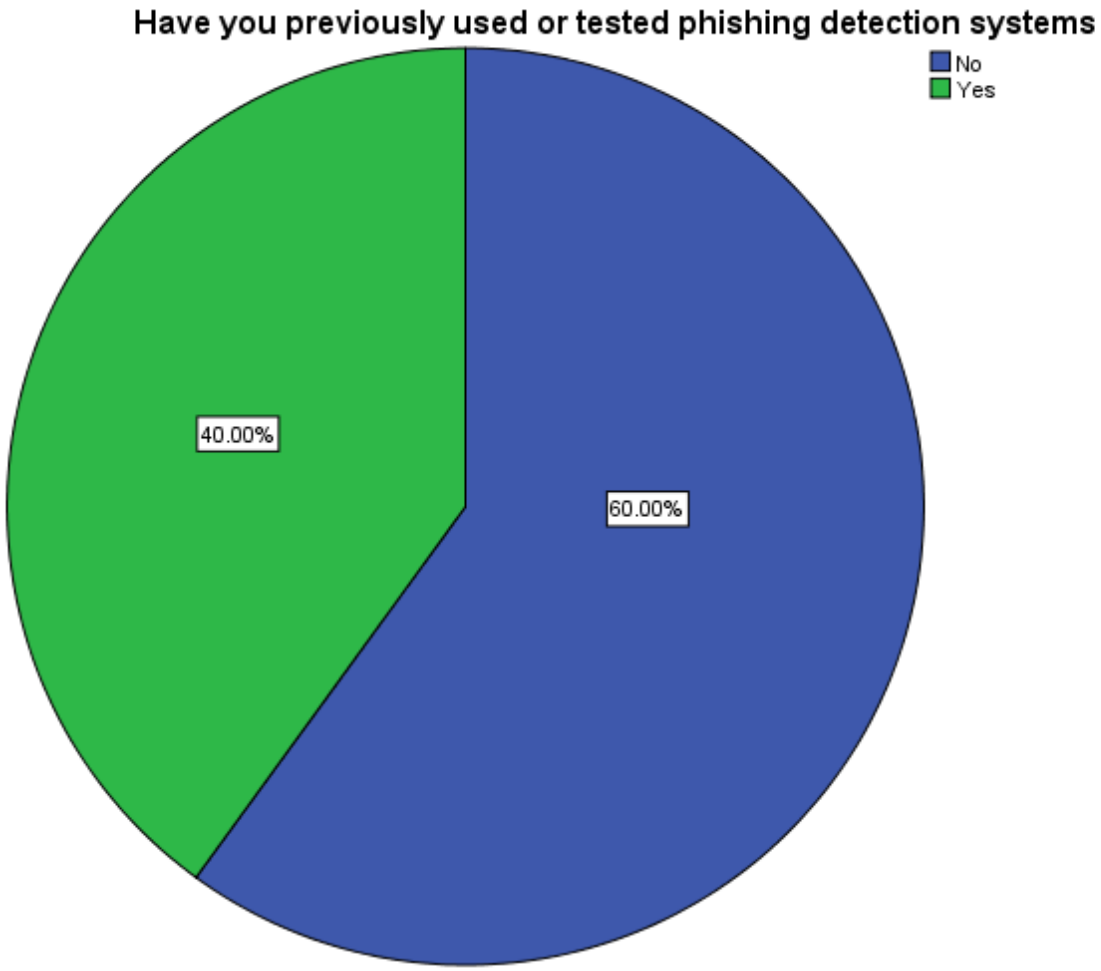

Figure 4-12 prior use of detection Model

The pie chart shows how much experience respondents have had with phishing detection systems. It shows that 60% of the people surveyed have never used or tested these systems before, while 40% have some prior experience. This suggests that most of the participants aren't very familiar with phishing detection technology.

4.11.2.6. Analysis of Model –Related Likert-Scale Results

The survey respondents rated 14 statements concerning the model functionalities ,effectiveness and user experience on a scale of 1 to 5 Likert scale which included the following: (1:Strongly Disagree to 5 : Strongly Agree).

Table4-14 Summary Statistics

	N	Minimum	Maximum	Mean	Std. Deviation
The model efficiently analyzes multiple email features (headers, content, subject, URLs).	20	1	5	3.90	1.021
The model architecture (CNN + LSTM) provides better accuracy than traditional ML models."	20	1	5	3.95	1.050
The training data from Kaggle Data are appropriate and reliable for phishing detection.	20	2	5	4.25	.967

The model achieves consistent accuracy and minimal overfitting after multiple epochs	20	1	5	3.80	1.056
The confusion matrix results show strong predictive power and reliability.	20	2	5	3.65	.813
The use of CNN for feature extraction and LSTM for sequence learning improves detection performance.	20	1	5	3.65	1.040
The system can be applied in real-world email filtering and cybersecurity operations.	20	1	5	4.15	1.040
The Flask API interface is easy to use and understand.	20	1	5	3.65	1.089
The system provides responses quickly without significant delay.	20	1	5	3.80	.951
The system can be easily integrated into existing email platforms or servers.	20	1	5	3.70	.979

The instructions provided on the system interface are sufficient.	20	1	5	3.80	1.005
The system enhances user confidence in identifying safe and unsafe emails.	20	2	5	4.15	.875
Overall, the system provides a satisfactory user experience.	20	1	5	4.25	1.118
Valid N (listwise)	20				

4.12. Results

The analysis of the number of respondents (n=20) gives a positive assessment of the CNN-LSTM phishing detection system. The findings suggest that the users tend to agree that the model is effective in analyzing various characteristics of emails, the mean of which is 3.90. Also, the respondents assume that the hybrid CNN -LSTM architecture would give better accuracy than the classic machine learning models, which is depicted by a mean of 3.95. The reliability and suitability of training data as a source used in Kaggle is also one of the strongest aspects that were highlighted in the evaluation with a high mean score of 4.25 indicating that the user has high confidence in the quality of the datasets. Moreover, the users will agree that the model will consistently be accurate with little overfitting when the number of training epochs increases but the standard deviation (1.056) is very high implying different experiences across the participants.

The confusion matrix was seen to exhibit a good predictive ability in terms of model performance

and interpretation of output and thus attained an average of 3.65 with a fair amount of variation in responses. On the same note, the opinion of the users on the matter that a combination of CNN in feature extraction and LSTM in sequence learning was significantly superior to that of detection was moderately high though the opinion varied here a little as indicated by the standard deviation. Usability Practical usability of the system was also well received. The respondents also greatly believed that the system is applicable to the real life scenario of cybersecurity and email filtering with an average score of 4.15 which is a strong indicator of practical confidence. The Flask API interface was rated moderately as it has a mean score of 3.65 with many users agreeing that it was easy to use with a few seeing the need to improve the user interface.

Other performance related features like speed of the system were also considered to be positive with the respondents showing that the system reacts in a timely manner without much delays as indicated by a mean of 3.80. Ability to integrate also ranked high (mean = 3.70), indicating that the users are of the opinion that the system can be integrated into the existing email systems, with some uncertainty being observed. The users were also generally in agreement that the instructions were adequate with a mean of 3.80 though the variation showed that there may be an opportunity of more directive instructions. Notably, the system was observed to increase the user confidence in the recognition of safe and unsafe emails (mean = 4.15), which indicates the positive effect on the user.

Lastly, overall user experience was the most rated with an average of 4.25. This implies that although there are still a few things that require improvement, like simplification of interfaces, clarity of instructions, and a requirement to have a consistent model performance, users are

satisfied with the system, and consider it to be effective and useful. Taken as a whole, the findings confirm a high degree of the trustworthiness of the system, its ease of use, and practicality in the real-life experience, and performance consistency can be improved further.

4.13. Theoretical contributions

Theoretical contribution of this study makes significant contributions to the body of knowledge in several critical areas which include the feature engineering fusion theory and the demonstration use of the interface.

This study builds on earlier phishing detection work by combining URL and text data into one deep learning system called CNN-LSTM. Studies that only used manually created features or standard machine learning methods, this research uses the raw URL or text itself as the main input. This lets the model learn important patterns like meaning, structure, and word order directly from the data instead of relying on features like URL length or number of dots. The study also improves on past methods by mixing two techniques: CNN, which finds small patterns in the URL or text, and LSTM, which understands the full context by looking both forward and backward. The main new part of this study is using all these steps together in one system for phishing detection. This goes beyond previous methods that used separate features, traditional classifiers, or single deep learning models. It shows how both local and sequential patterns can be learned together from URL or text data to help detect phishing more effectively.

The interface of the model was user-friendly, and even people who have no previous experience on phishing detection can work with the system without any doubts, which makes the model fit the real-world organizational setting where non-technical personnel are frequently targeted by the

attackers. The system has the potential to strongly mitigate the number of phishing messages accessed by employees as a pre-filtering tool, which enhances organizational resiliency to social engineering attacks. Also, the fact that the tool is used to reveal malicious indicators makes it useful in cybersecurity awareness and training, where it can contribute to enhancing the digital literacy of users by showing how phishing components manifest themselves in email messages. Lastly, the model provides a basis for future improvements, allowing those developing it to expand its features to multilingual email analysis, image-based phishing detection, more advanced explainable AI modules, or real-time email scanning, therefore making sure that the system could evolve alongside the ever evolving threat platform.

4.14. Practical contributions

The practical value of the study was much broader than its academic value, as it offers highly applicable results that organizations, software developers, cybersecurity experts, and system administrators can implement to improve their security infrastructure. The CNN-LSTM phishing detector developed in this work can be fully implemented in email systems, Security Information and Event Management (SIEM) systems, and cloud-based cybersecurity systems, providing a viable solution that organizations can adopt to enhance their defense against phishing attacks.

Practical deployment with its use of a Flask API can be further supported because it can be easily integrated into existing enterprise systems without needing to re-engineer or significantly change their architecture, which is cost-effective and operationally convenient. Moreover, the system has a convenient and easy-to-use interface, thus allowing even those with a history of no previous experience with phishing detection or cyber security software to navigate and operate the system

easily, which is vital in the actual organizational setting where the majority of end-users are not technical.

The model itself is helpful in enhancing the organizational email security, as it acts as a pre-filter automatically recognizing and labeling a potential phishing email that is sent to the employees to block the chances of a successful social engineering attack considerably. In addition to direct protection, the system may have an important educational role; it may be included in the cybersecurity awareness and training program to educate users on identifying signs of phishing by pointing out particular malicious factors in emails, which effectively enhances the digital literacy and the culture of security in the organization.

Moreover, this system provides the solid base of future improvements, providing developers with the opportunity to develop the model further, to add multilingual email analytics, to identify image-based phishing attacks, or to run an operation within the real-time scanning mode, making the model rather adaptive and applicable in the context of ever-evolving phishing and other cyber-attacks.

4.1.5 Methodological Contribution

This study introduces a new CNN-LSTM hybrid model for phishing detection that automatically extracts features and learns from the sequential structure of emails and URLs.

Unlike older machine learning methods that rely on manually engineered features, this approach learns directly from data, making it stronger, more scalable, and better at adapting to current and future phishing threats.

4.14. Challenges of CNN-LSTM model over other Models

Although CNN-LSTM models excel in their text classification activities (e.g. phishing emails) in comparison to simple machine learning models or single-model deep learning models, they are associated with various challenges. To begin with, CNN -LSTM models are computationally expensive due to the use of convolutional processes to extract features, and recurrent layers to learn sequences. This two-architecture model needs additional memory, additional training time and more powerful hardware than simpler machine learning models, such as Random Forest, SVM, or Logistic Regression, and even simpler deep networks, such as standalone CNN or LSTM models. The other problem is the hyper-parameters tuning; it is harder to choose an appropriate number of filters, kernel sizes, sequence lengths, hidden units, dropout rates, and learning rates when two deep learning models are combined.

Also CNN-LSTM models are also vulnerable to overfitting on small or unbalanced data sets, since when learning a large number of parameters the model is highly likely to be over-fitted. This is usually costly in terms of regularization or huge datasets to be carried out. It is also harder to interpret the decisions of hybrid deep learning models and thus it is not easy to explain the predictions as compared to the tree-based models or single-layer neural networks. Lastly, the application of CNN-LSTM models in real-time systems may cause latency and integration issues since the model will require both convolutions and sequential dependencies to be calculated to make this prediction. Such aspects render CNN-LSTM models more challenging and complicated in comparison with most other methods.

CHAPTER FIVE
SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1. Introduction

The section provides the final overview of the paper, the research on the creation and testing of CNN-LSTM phishing detector. It provides an overview of the main results, emphasizes the importance of the hybrid deep learning architecture, and reconsiders the way the model will be relevant to enhancing the phishing email detection in the contemporary cybersecurity context. The section also explains the analysis outcome of the tests conducted by experts on the effectiveness, reliability and usability of the proposed system through evaluation of the system performance measurement. Lastly, it offers real world suggestions that are intended to enhance the model, its application in the actual email infrastructure, and future research directions on more advanced and scalable AI-based phishing detection systems.

5.2. Summary

Phishing attacks are more advanced, and the attackers are posing as trusted wholly and manipulated text or shortened text to gain access to sensitive information. Conventional machine learning models like the Support Vector Machines (SVM) and decision trees are highly dependent on the manual feature extraction involving the security experts in developing features that include the length of the URLs, the number of dots and special characters. This will render them less useful when it comes to tracking changing trends in phishing in real-time.

Most of the past literature has been predominantly dealing with character-level URL information and has not concerned itself much with the semantic interpretation of domain names, subdomain,

and query terms. Consequently, such strategies find it difficult to identify legitimate and malicious emails. The use of Email obfuscation by using instant messaging systems and Business Email Compromise (BEC) phishing techniques also make it difficult to detect since such attackers use obfuscated links that are shortened or layered redirections and cannot be detected by conventional filters.

To overcome these shortcomings, this paper suggests a hybrid CNN-LSTM. The CNN element derives local patterns and structural information of the URLs to identify obfuscation and shortened URLs and the LSTM network derives sequential context information to comprehend more deep semantic relationships among the URL tokens. The architecture is enhanced by detection of phishing URL by varying formats and environment.

The end goal of the research is to compare CNN-LSTM with such classical models like decision trees and random forests and in the context of the phishing emails, recognize shortened and obfuscated URLs. The idea is to come up with a powerful system that is able to adapt to changing phishing threats.

5.3 Conclusion

In conclusion, the design, deployment, and professional analysis of the CNN-LSTM-based phishing detection model was a tremendous leap in the field of cybersecurity using artificial intelligence, and the description of its relevance can highlight the extent of its contribution. The very creation of this hybrid system, which was based on the complementary advantages of the Convolutional Neural Networks and the Bidirectional Long Short-Term Memory networks, shows

that deep learning keeps improving to remain the potent tool against more sophisticated phishing threats that traditional models cannot identify.

This goal was reached by looking at 28 journal articles that were picked from 40 found using Google Scholar. The review included models like SVM, Decision Trees, Random Forest, KNN, Neural Networks, NLP, LSTM, CNN, and hybrid models. The results were compared in Table 4.1, which listed each technique, the dataset used, their strengths and weaknesses, and the accuracy reported. The analysis found that traditional machine learning models rely on manually creating features. They are also likely to overfit and have trouble with tricky, shortened, and changing phishing URLs. Deep learning and hybrid models like CNN, LSTM, and CNN-LSTM can automatically learn features and pick up both spatial and sequential patterns. However, these models use a lot of computing power and need big datasets. This gap in existing research led to the decision to use a hybrid CNN-LSTM approach.

This goal was reached by turning the strengths from Objective 1 into a suggested model structure (Figure 4.2) and its math equations (Equations 4.1 to 4.7).The model uses an embedding layer, a convolutional layer with max pooling to find local patterns, and a Bidirectional LSTM to understand the context from both directions. During training, binary cross-entropy loss is used along with the Adam optimizer. The whole process includes collecting data from Kaggle, preparing the data, extracting features, training the model with a 70:30 split, evaluating its performance, deploying it using Flask, and setting up continuous learning. The final design eliminates the need for manual feature engineering and combines the ability of CNN to detect patterns with the sequence-learning power of LSTM.

This goal was reached by using the designed model on a Kaggle dataset that had 8,986 records, evenly split between 4,493 real emails and 4,493 phishing emails. The model was trained for five cycles, and during this time, the validation accuracy went up from 57.09% to 91.48%, while the loss kept decreasing. The model ended up with an accuracy of 97.88%, precision of 90.85%, recall of 95.02%, an F1-score of 92.89%, and a ROC-AUC of 0.9798.The confusion matrix showed only 144 false positives and 75 false negatives. A filter size of 24 gave the best results compared to sizes 8 and 16.The model performed better than the baseline models listed in Table 4.7, such as SVM (93%), Decision Tree (96%), KNN (87%), Neural Networks (93%), and Random Forest (96%).It was turned into a working Flask web interface that can classify incoming email as either phishing or legitimate.

This goal was reached by having 20 carefully selected experts in fields like computer science, cybersecurity, information technology, and data science evaluate the system. Most of these experts had at least a bachelor's degree, some had master's or PhDs, and the group included both new and experienced users. After seeing a demonstration of the system, the experts filled out a 14-question survey with five possible answers for each question.

The ratings showed that overall user experience and the reliability of the training data were the highest, with an average score of 4.25 each.

Next were the effectiveness of the CNN-LSTM model compared to traditional models (3.95) and the analysis of email features (3.90).The system's responsiveness and clear instructions scored 3.80, while integration scored 3.70 and the ease of understanding confusion matrices scored 3.65.

A correlation analysis using SPSS found strong, significant positive links between model

reliability, the model's architecture, system integration, and user satisfaction. Additionally, gender did not have a noticeable impact on how the system was viewed.

This broadened meaning of the conclusion demonstrates that the demographic analysis used in this analysis was essential in the evaluation process, due to the fact that the experts used to test the system were diverse in terms of education, specialization, and experience level, hence the feedback and assessments were representative and not restricted to a small group of professionals. The majority of such experts were well-educated, and most had earned Bachelor's, Master's, or even PhD degrees, with a solid background in computer science, cybersecurity, data science, and information technology.

This diversity made their evaluations more credible, since participants with technical backgrounds would experience the model with detailed insight into its architecture, performance, and learning behavior, whereas those with cybersecurity knowledge would offer a realistic view of the model's functioning in real-world threat contexts. Such a mix of novice and expert participants further enhanced the evaluation: less experienced participants evaluated the system in terms of usability, clarity, and user-intuitive interface, whereas more experienced respondents criticized the technical features, enabling an evaluation of the system as a whole from both user-oriented and expert-oriented perspectives.

When analyzing the Likert-scale data, it can be concluded that the model was highly rated in all the assessment dimensions. The strength of the feature extraction and sequence learning processes was always confirmed by the experts who stated that CNN-LSTM architecture could be used to

analyze various features of emails, such as headers, subjects, URLs, and even body contents, which supports the effectiveness of the algorithm. It is especially significant since in phishing attacks the manipulated context and structural details can be easily identified, which is not easy with the conventional machine learning algorithms. The fact that the architecture can extract meaning based on both the local features patterns (treated by the CNN) and long-range textual dependencies (treated by the LSTM) means that the architecture has a complex view of email content, and it is much better able to differentiate legitimate communication and malicious communication.

Also, the high data reliability rating indicates that training the model on Kaggle data made it more authentic and relevant. Actual phishing emails and URLs enable the system to learn from real attack patterns, enhancing its ability to generalize in real-life settings. The Likert-scale findings also revealed that the model was found to be very stable and had very little overfitting as training epochs went on, but there were variations in the expert opinion, which could be refined further by tuning hyperparameters or regularizing the model.

However, it was generally accepted that the model is reliable enough to be relied upon in practice. The predictive reliability was also proved to be critical as another essential factor that has been established by the evaluation of the confusion matrix. This matrix plays a significant role in identifying the effectiveness of the model in differentiating between phishing and legitimate emails and the positive feedback is indicative that CNN-LSTM system could achieve an acceptable balance between true positives and true negatives with a minimum of misclassifications. This is fundamental, as false positives might interfere with workflow, as they might label safe emails as dangerous when they do not need it, whereas false negatives might provide users with an access

to dangerous phishing emails. The high performance meant that the model is competent enough to reduce both forms of errors hence helping to make the communication environment safer.

Another factor in the successful evaluation of the system was usability. The professionals stated that the system's interface was rather straightforward, as it was built with Flask; however, it was easy for the end-user to navigate and use without a clear technical background. This is critical to cybersecurity tools, as usability determines a system's acceptability to end users. A tool that is technically strong and hard to operate can hardly be widely adopted, whereas CNN-LSTM model in the framework of the study effectively achieved the balance between technical complexity and easy-to-use design.

The high scores in general user experience and user confidence also demonstrate the need to design a system that allows users to feel more empowered and better protect the safety of their emails. It is also vital to acknowledge that the model may be implemented in real-world cybersecurity ecosystems. When the scores of system integration and real-world applicability are positive, it shows that the architecture and design of the model can be used in real-life infrastructures, such as within an email server, email gateway, or cloud-based filtering systems. The system, with some minor improvements, might be used as an added protection layer that automatically filters incoming emails and provides users with warnings about potential threats, considerably minimizing the risk of phishing attacks at the organizational level.

The research contribution before the theory is even more significant when considered further. The research improves current understanding of optimizing deep learning models in cybersecurity

activities by designing and experimenting with a hybrid CNN-LSTM model to detect phishing attacks. It shows that a system combining convolutional feature extraction and sequence text modeling can identify more phishing indicators than any single-architecture model. This confirms and expands the existing literature, which has supported hybrid models but has not been thoroughly assessed in email-based settings.

Practically, the research has important implications to the cybersecurity fraternity. The model created in this paper can be implemented as a practical phishing detector that could assist companies in decreasing the number of human errors as this is the primary cause of successful cyberattacks. It can enhance email security privileges, lessen the strain on the security departments due to the automated examination of suspect emails, and enhance the awareness of the end-users by offering transparent indications of the type and potential intrusions. Also, the Flask-based interface, although it is minimal, forms the basis of future development of graphical dashboards, real-time alerting system, and integration tools that might improve the usability and the adoption prospects of the system.

The limitations found in the model that involves the requirement to use larger datasets, better interface design, and model consistency can be viewed as advantages and not weaknesses but can be improved with time. Such constraints are characteristic of the developing AI-based cybersecurity tools, particularly those that are being created in academia. The generalization capability of the model would be enhanced by increasing the dataset size, particularly in the detection of a new form of phishing attacks.

The broadened description of the conclusion holds tenacity to the fact that the hybrid CNN-LSTM model is one of the most effective solutions to the problem of phishing the most widespread and dangerous cybersecurity threat of the digital epoch. The research shows that deep learning is not only an additive tool but also that it forms the basis of contemporary cybersecurity defense, through theoretical development and real applications. These results confirm the potential of AI-based email security solutions and open the prospect of creating more intelligent, scalable, and adaptable phishing-protection models for users, organizations, and digital systems on a global scale.

5.4. Recommendations

According to the document, one can derive several clear recommendations to improve and deploy the CNN-LSTM phishing detection model in the future. The first step is to integrate the system with live email services, such as enterprise SMTP servers, Microsoft Exchange, or Gmail APIs, to test its viability under real-world conditions and ensure it is reliable for handling live email traffic. Also, more sophisticated hyperparameter optimization methods, such as Bayesian optimization, grid search, or genetic algorithms, are suggested to enhance model stability, minimize variability in model performance, and improve predictive accuracy.

Individually, to further reduce false positives and false negatives, the implementation of ensemble learning strategies based on CNN-LSTM models and transformer or attention-based architectures is recommended. Extending the model to multilingual phishing detection, particularly for languages frequently used by attackers such as Swahili, French, and Arabic, would greatly expand its utility and strength. According to the usability angle, an opportunity would be to upgrade the existing Flask-based interface to a new front-end system like react to enhance user experience, navigation, and visual analytics.

Lastly, the integration of Explainable AI methods, e.g., LIME, SHAP, or attention visualization, would make the system more user-friendly or operational, as well as improve user trust and insights into the phishing patterns, which would be a more suitable tool in operational use and training programs on cybersecurity.

REFERENCES

Abraham, S., & Patel, D. (2019). Phishing Attacks: An Analysis of User Behavior and Malicious Link Detection. *International Journal of Information Security,* 21(4), 295-304.

Adebowale, M. A., Lwin, K. T., & Hossain, M. A. (2023). Intelligent phishing detection scheme using deep learning algorithms. *Journal of Enterprise Information Management*, *36*(3), 747-766.

Adebowale, A., et al. (2020). Phishing email detection using natural language processing techniques. *Journal of Cybersecurity and Privacy*, 1(1), 60–79.

Ahmed, D. S., Hussein, K. Q., & Allah, H. A. A. A. (2022). Phishing websites detection model based on decision tree algorithm and best feature selection method. *Turkish Journal of Computer and Mathematics Education*, *13*(1), 100-107.

Ahmed, N., Amin, R., Aldabbas, H., Koundal, D., Alouffi, B., & Shah, T. (2022). Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges. *Security and Communication Networks*, *2022*(1), 1862888.

Akinyelu, A. A., Adewumi, A. O., et al. (2023). *A systematic review on deep-learning-based phishing email detection*. Electronics, 12**(**21), 4545.

Alabdan, R. (2020). Phishing attacks survey: Types, vectors, and technical approaches. *Future internet*, *12*(10), 168.

Aleroud, A., Alsmadi, I., & Damiani, E. (2022). Evaluating Human and Machine Performance in Phishing Email Detection: A Signal Detection Approach. *Computers & Security*

Aldallal, A. (2022). Toward an efficient intrusion detection system using hybrid deep learning approach. *Symmetry*, *14*(9), 1916.

Alhaj, A., et al. (2023). *A comprehensive dual-layer architecture for phishing and spam email detection*. Computers & Security, 133**,** 103378.

Aljofey, A., Jiang, Q., Qu, Q., Huang, M., & Niyigena, J.-P. (2020). An Effective Phishing Detection Model Based on Character-Level Convolutional Neural Network from URL. *Electronics*, *9*(9), Article 9. https://doi.org/10.3390/electronics9091514

Alom, M., Carminati, B., & Ferrari, E. (2020). *Spam detection in Twitter using Convolutional Neural Networks and metadata fusion*. *Journal of Information Security and Applications, 54***,** 102529.

Alsariera, Y. A., Adeyemo, V. E., Balogun, A. O., & Alazzawi, A. K. (2020). AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites. *IEEE Access*, *8*, 142532–142542. IEEE Access. https://doi.org/10.1109/ACCESS.2020.3013699

Alshaikh, M., Maynard, S. B., Ahmad, A., & Chang, S. (2021). Cybersecurity culture in organizations: A review of research, definition, indicators, and suggestions for future research. *Computers & Security, 102*, 102142.

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.

Altwaijry, N., Al-Turaiki, I., Alotaibi, R., & Alakeel, F. (2024). *Advancing phishing email detection: A comparative study of deep learning models*.

Alzahrani, A., & Alenazi, M. (2021). A hybrid deep learning model for phishing email detection using convolutional and recurrent neural networks. *IEEE Access*, *9*, 112291–112304.

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: Concepts,

CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1),

An, H., Ma, R., Yan, Y., Chen, T., Zhao, Y., Li, P., & Lv, C. (2024). Finsformer: A novel approach to detecting financial attacks using transformer and cluster-attention. *Applied Sciences*, *14*(1), 460.

Andriu, A. V. (2023). Adaptive phishing detection: Harnessing the power of Artificial Intelligence for enhanced email security. *Romanian Cyber Security. J*, *5*(1), 3-9.

Ariyadasa, S., Fernando, S., & Fernando, S. (2020). Detecting phishing attacks using a combined model of LSTM and CNN. *Int. J. Adv. Appl. Science*, *7*(7), 56-67.

Asiri, S., Xiao, Y., Alzahrani, S., Li, S., & Li, T. (2023). A survey of intelligent detection designs of HTML URL phishing attacks. *IEEE Access*, *11*, 6421-6443.

Atlam, H. F., & Oluwatimilehin, O. (2023). Business email compromise phishing detection based on machine learning: A systematic literature review. *Electronics*, *12*(1), Article 42.

Atawneh, S., & Aljehani, H. (2023). Phishing Email Detection Model Using Deep Learning. *Electronics*, *12*(20), Article 20. https://doi.org/10.3390/electronics12204261

Bahnsen, A. C., et al. (2017). Detecting phishing emails using supervised learning algorithms. In *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*.

Basit, A., Zafar, M., Javed, A. R., & Jalil, Z. (2020). A novel ensemble machine learning method to detect phishing attack. In *2020 IEEE 23rd International Multitopic Conference (INMIC)* (pp. 1-5). IEEE.

Bauskar, S. R., Madhavaram, C. R., Galla, E. P., Sunkara, J. R., & Gollangi, H. K. (2024). AI-driven phishing email detection: Leveraging big data analytics for enhanced cybersecurity. *Library Progress International*, *44*(3), 7211-7224.

Blancaflor, E., Calpo, A. H., Cebrian, S. J., & Siquioco, F. (2024). A Comprehensive Review of

Neural Network-Based Approaches for Predicting Phishing Websites and URLs. In *2024 5th International Conference on Industrial Engineering and Artificial Intelligence (IEAI)* (pp. 96-101). IEEE.

Brownlee, J. (2021, April 18). A Gentle Introduction to Ensemble Learning Algorithms. *MachineLearningMastery.Com*.https://www.machinelearningmastery.com/tour-of-ensemble-learning-algorithms

Bu, S.-J., & Cho, S.-B. (2021). Deep character-level anomaly detection based on a convolutional autoencoder for zero-day phishing URL detection. *Electronics, 10*(12), Article 1492.

Communications Authority of Kenya. (2023). *National KE-CIRT/CC cybersecurity report*. https://www.ca.go.ke

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation.

Datta, P., Panda, S. N., Tanwar, S., & Kaushal, R. K. (2020). A Technical Review Report on Cyber Crimes in India. *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, 269–275. https://doi.org/10.1109/ESCI48226.2020.9167567

Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union. (2022). *Official Journal of the European Union, L 333*, 80–152. https://eur-lex.europa.eu/eli/dir/2022/2555/oj

Do, X. C., Nguyen, H. A., & Tran, N. H. (2022). Malicious URL detection based on deep learning techniques using character-level and word-level embeddings. In *Proceedings of the International Conference on Artificial Intelligence and Knowledge Engineering Applications*

Editor. (2024, January 18). *Kenya's Cyber Security in the Digital Landscape—Africa.com*.

https://www.africa.com/kenyas-cyber-security-in-the-digital-landscape/

Eze, C. S., & Shamir, L. (2024). Analysis and prevention of AI-based phishing email attacks. *Electronics*, *13*(10), 1839.

Fares, H., Kilani, J., Fagroud, F., Toumi, H., Lakrami, F., Baddi, Y., & Aknin, N. (2024). Machine learning approach for email phishing detection. *Procedia Computer Science*, *251*, 746-751.

Feng, Q., Zhou, Y., Fan, J., & Wang, W. (2018). *Detecting spam URLs in social media via convolutional neural networks and word embeddings*. IEEE Access, 6, 13858–13867.

Gajjar, M., & Taherdoost, H. (2024). *Global phishing threat report 2024*. ThreatLabz.

Hadlington, L., Shotton, M., & Parsons, K. (2019). Who is most at risk? A psychometric assessment of the everyday cyber risks faced by older adults. *Behavior & Information Technology, 38*(8), 776–786.

Hussain, M., Cheng, C., Xu, R., & Afzal, M. (2023). *CNN-Fusion: An effective and lightweight phishing detection method based on a multi-variant ConvNet***.** Information Sciences, 631**,** 328–345.

Interpol. (2025). *Africa cyber threat assessment report*. https://www.interpol.int/

Jain, A. K., & Gupta, B. B. (2019). A machine learning-based approach for phishing detection using hyperlink information. *Journal of Ambient Intelligence and Humanized Computing*, *10*, 2015-2028.

Kavya, S., and D. Sumathi. "Staying Ahead of Phishers: A Review of Recent Advances and Emerging Methodologies in Phishing Detection." *Artificial Intelligence Review* 58, no. 2

KE-CIRT/CC. (2024). *Cybersecurity threat landscape reports and advisories*. https://www.ke-cirt.go.ke

Kim, Y., Lee, S., & Kim, H. (2020). A deep learning-based approach for detecting phishing URLs.

Computers & Security, 89, 101674.

Kshetri, N. (2019). Cybercrime and Cybersecurity in Africa. Journal *of Global Information Technology Management*, *22*(2), 77–81.

Korkmaz, S., Arslan, M., & Doğru, İ. A. (2021). Detecting phishing websites using deep learning and convolutional neural networks. *Neural Computing and Applications*, *33*, 11227–11238.

Kritika. (2024). A comprehensive literature review on phishing URL detection using deep learning techniques. *Journal of Cyber Security Technology*, *9*(4), 315-343.

Kyaw, P. H., Gutierrez, J., & Ghobakhlou, A. (2024). A systematic review of deep learning techniques for phishing email detection. *Electronics*, *13*(19), 3823.

Jamal, S., Wimmer, H., & Sarker, I. H. (2024). An improved transformer-based model for detecting phishing, spam and ham emails: A large language model approach. *Security and Privacy*, *7*(5), e402.

Khatri, K. K. (2020). Research paradigm: A philosophy of educational research. *International Journal of English Literature and Social Sciences, 5*(5), 1435-1440.

Madhavaram, C., Konkimalla, S., Rajaram, S. K., Gollangi, H. K., & Reddy, M. (2023). *AI/ML-Powered Phishing Detection: Building an Impenetrable Email Security System*. 10–19.

Marcum, C. D., Higgins, G. E., & Ricketts, M. L. (2020). *Cybercrime victimization among young adults: An examination of risk factors and gender differences*.

Meléndez, R., Ptaszynski, M., & Masui, F. (2024). Comparative Investigation of Traditional Machine-Learning Models and Transformer Models for Phishing Email Detection. *Electronics*, *13*(24), 4877.

Maneriker, P., Stokes, J. W., Lazo, E. G., Carutasu, D., Tajaddodianfar, F., & Gururajan, A. (2021, November). Urltran: Improving phishing url detection using transformers. In *MILCOM*

2021-2021 IEEE Military Communications Conference (MILCOM) (pp. 197-204). IEEE.

Nanda, M., & Goel, S. (2024). URL based phishing attack detection using LSTM-gated highway attention block convolutional neural network. *Multimedia Tools and Applications*, *83*(27), 69345-69375.

Nasreen, G., Khan, M. M., Younus, M., Zafar, B., & Hanif, M. K. (2024). Email spam detection by deep learning models using novel feature selection technique and BERT. *Egyptian Informatics Journal*, *26*, 100473.

National Institute of Standards and Technology [NIST]. (2024). *The NIST cybersecurity framework (CSF) 2.0* (NIST CSWP 29). https://doi.org/10.6028/NIST.CSWP.29

Rathee, T., & Mann, P. S. (2022). Performance evaluation of deep learning models for phishing email detection**.** Journal of Information and Knowledge Management, 21**(**04), 2250061.

Rezigalla, A. A. (2020). Observational study designs: Synopsis for selecting an appropriate study design. *Cureus*, *12*(1), Article e6692.

Safi, A., & Singh, S. (2023). A systematic literature review on phishing website detection techniques. *Journal of King Saud University - Computer and Information Sciences, 35*(2

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A systematic literature review on phishing email detection using natural language processing techniques. *Ieee Access*, *10*, 65703-65727.

Sakthipriya, N., Govindasamy, V., Abhinesh, C., Govarthini, K. T., Rajesh, R., & Swedha, R. (2024). *Enhancing phishing detection: A hybrid deep learning model integrating LSTM and Bi-GRU algorithms for URL and content analysis*. International Education and Research Journal (IERJ), 10**(3).**

Shahariar, M. S., Ahmed, M., & Nur, M. M. (2019). *Spam detection in reviews using deep*

learning. In *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)* (pp. 1–5). IEEE.

Slam, M. R., Abawajy, J. H., & Watters, P. A. (2022). *A hybrid deep learning and KNN model for phishing detection*. Journal of Information Security and Applications.

Su, Y., & Su, J. (2023). Phishing URL detection using BERT-based contextual representations. *Sensors, 23*(20), 8499.

Opara, C., Wei, B., & Chen, Y. (2019). *HTMLPhish: Enabling phishing web page detection by applying deep learning techniques on HTML analysis*.

Ozcan, A., Catal, C., Donmez, E., & Senturk, B. (2023). A hybrid DNN–LSTM model for detecting phishing URLs. *Neural Computing and Applications, 35*(7), 4957–4973.

Peng, Y., Zhang, X., Xu, S., & Li, W. (2019).A malicious URL detection model based on CNN-LSTM with attention mechanism.IEEE Access, 7, 146389–146398.

Phan, T. Q., Bui, H. D., & Nguyen, T. T. (2020). *Phishing URL detection using deep learning techniques*. Proceedings of the International Conference on Artificial Intelligence.

PhishRepo Consortium. (2022). *PhishRepo: A seamless collection of phishing data to fill a research gap*. *The Scientific Association for Information*.

Piñeiro, J. L., & Portillo, L. W. (2022). Web architecture for URL-based phishing detection based on Random Forest, Classification Trees, and Support Vector Machine. *Inteligencia Artificial, 25*(69), 107–121.

Rana, J., Dilshad, S., & Ahsan, Md. A. (2021). *Ethical Issues in Research* (pp. 1–7).

Rao, R. S., & Pais, A. R. (2020). *Detection of phishing websites using an efficient feature-based machine learning framework*. Neural Computing and Applications, **32**, 479–494.

Raza, M., Jayasinghe, N. D., & Muslam, M. M. A. (2021). A Comprehensive Review on Email

Spam Classification using Machine Learning Algorithms. *2021 International Conference on Information Networking (ICOIN)*, 327–332.

Rhanoui, M., Mikram, M., Yousfi, S., & Barzali, S. (2019). A CNN-LSTM model for document-level sentiment analysis. *Machine Learning and Knowledge Extraction*, *1*(3), 832-847.

Roy, S. S., Awad, A. I., Amare, L. A., Erkihun, M. T., & Anas, M. (2022). Multimodel phishing url detection using lstm, bidirectional lstm, and gru models. *Future Internet*, *14*(11), 340.

Safi, S., & Singh, M. (2023).Malicious URL detection using machine learning techniques: A comparative analysis.International Journal of Advanced Computer Science and Applications, 14(1), 324–332.

Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. *Expert Systems with Applications*, *117*, 345–357.

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques. *IEEE Access*, *10*, 65703–65727. IEEE Access. https://doi.org/10.1109/ACCESS.2022.3183083

Sarker, I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Saxe, J., & Berlin, K. (2017). *Deep neural network-based malware detection using two-dimensional binary program features*. In MALWARE.

Seth, J., & Biswas, A. (2017). *Multimodal deep learning approach for image and text based spam detection*. In *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)* (pp. 1–4). IEEE.

Shombot, E. S., Dusserre, G., Bestak, R., & Ahmed, N. B. (2024). An application for predicting

phishing attacks: A case of implementing a support vector machine learning model. *Cyber Security and Applications, 2*, 100036.

Singh, R., Kumar, R., & Singla, R. K. (2020). A heuristic-based phishing detection approach using machine learning techniques. *International Journal of Information Technology, 12*(3), 685–690.

Sirigineedi, P., Bhattacharya, P., & Chakraborty, R. S. (2020). PhishAri: A hybrid model for phishing website detection using CNN and LSTM. *Procedia Computer Science, 167*, 1048–1057.

Srinivas, J., & Kumar, A. (2019). *Cybersecurity: Threats, attacks, and defense mechanisms*. International Journal of Computer Science and Information Security, 17**(5), 120**–128.

Standard, B. (2024, April 30). *India ranks third globally for phishing attacks after US, UK: Report*. https://www.business-standard.com/india-news/india-ranks-third-globally-for- phishing-attacks-after-us-uk-report-124043001165_1.html

Taloba, A. I., & Ismail, S. S. I. (2019). An Intelligent Hybrid Technique of Decision Tree and Genetic Algorithm for E-Mail Spam Detection. *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 99–104.

Thakur, K., Ali, M. L., Obaidat, M. A., & Kamruzzaman, A. (2023). A Systematic Review on Deep-Learning-Based Phishing Email Detection. *Electronics*, *12*(21), Article 21.

Uddin, M. A., Mahiuddin, M., & Sarker, I. H. (2026). An explainable transformer-based model for phishing email detection: A large language model approach. *Computer Networks*, 112061.

Verizon. (2025). *2025 data breach investigations report*. Verizon Business. https://www.verizon.com/business/resources/reports/dbir/

Verma, R., & Hossain, N. (2020). Feature engineering for phishing detection: A survey. *ACM Computing Surveys (CSUR)*, *52*(6), 1–28.

Vujović, Ž. Đ. (2021). Classification model evaluation metrics. *International Journal of Advanced Computer Science and Applications*, *12*(6), 599–606.

Wang, W., Zhang, F., Luo, X., & Zhang, S. (2019). PDRCNN: Precise phishing detection with recurrent convolutional neural networks. *Security and Communication Networks*, *2019*(1), 2595794.

Wang, D., et al. (2021). A hybrid deep learning model for phishing email detection. *Computers, Materials & Continua*, 67(3), 3455–3472.

Wei, F., Zeng, Y., Yang, Z., & Ma, H. (2020). Phishing detection based on deep learning and visual similarity. *IEEE Access*, *8*, 231423–231433.

Xiao, X., Xiao, W., Zhang, D., Zhang, B., Hu, G., Li, Q., & Xia, S. (2021). Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets. *Computers & Security*, *108*, 102372.

Xu, H., Zhou, Y., & Liu, Q. (2021). *Semantic spam detection in microblogs using a self-attention LSTM model with ALBERT embeddings*. Journal of Intelligent & Fuzzy Systems, 41(2), 3457–3467.

Yang, R., Zheng, K., Wu, B., Wu, C., & Wang, X. (2021). Phishing website detection based on deep convolutional neural network and random forest ensemble learning. *Sensors*, *21*(24), 8281.

Yaitskyi, A. (2022). Employing and improving machine learning of detection of Phishing URLs.

Zabihimayvan, M., & Doran, D. (2019). Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection. *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 1–

6. https://doi.org/10.1109/FUZZ-IEEE.2019.8858884

Zhu, E., Ju, Y., Chen, Z., Liu, F., & Fang, X. (2020). DTOF-ANN: An Artificial Neural Network phishing detection model based on Decision Tree and Optimal Features. *Applied Soft Computing*, *95*, 106505. https://doi.org/10.1016/j.asoc.2020.106505

Zuraiq, A. A., & Alkasassbeh, M. (2019). Review: Phishing Detection Approaches. *2019 2nd International Conference on New Trends in Computing Sciences (ICTCS)*, 1–6.

Zieni, R., Massari, L., & Calzarossa, M. C. (2023). Phishing or not phishing? A survey on the detection of phishing websites. *IEEE Access*, *11*, 18499-18519.

Zhao, C., Xin, Y., Li, X., Yang, Y., & Chen, Y. (2020). A Heterogeneous Ensemble Learning Framework for Spam Detection in Social Networks with Imbalanced Data. *Applied Sciences*, *10*(3), Article 3.

Zhang, Y., & Lee, D. (2019). Text-CNN for phishing email detection. *IEEE Access*, 7, 172933–172941.

Zhu, E., Ju, Y., Chen, Z., Liu, F., & Fang, X. (2020). DTOF-ANN: An Artificial Neural Network phishing detection model based on Decision Tree and Optimal Features. *Applied Soft Computing*, *95*, and 106505. https://doi.org/10.1016/j.asoc.2020.106505

Zhou, Y., & Cao, Y. (2019). A Survey of Phishing Detection Using Machine Learning and Signal Detection Theory. *ACM Computing Surveys*, 52(6), 1–35.

APPENDICES

APPENDIX 1: UNIVERSITY INTRODUCTORY LETTER

Kirinyaga University

Tel: +254 701562092, +254 728499650, +254 709742000/30
P.O. Box 143-10300 Kerugoya.

Email: info@kyu.ac.ke
Website: www.kyu.ac.ke

SCHOOL OF PURE AND APPLIED SCIENCES DEPARTMENT OF PURE AND APPLIED SCIENCES

The Chief Executive Officer National Commission for Science

Technology &Innovation (NACOSTI)

NAIROBI.

20th June, 2025.

Dear Sir/Madam

REF: RESEARCH PERMIT FOR MURIITHI NICHOLAS MURIUKI PA303/S/21176/23

The above subject refers.

The above named is a student of Kirinyaga University pursuing a degree course in PhD in Information Science in the Department of Pure and Applied Sciences. He has successfully defended his research proposal *"A Cyber Security-Based Convolution Neural Network Detection Model for Email Phishing Using Bi-Long Short Term Memory Algorithm."* fulfillment for the degree award.

We are kindly requesting your office to provide him with the necessary assistance for data collection and completion of his research.

Any assistance offered to the student will be highly appreciated.

Please to do not hesitate to contact the undersigned for any information regarding the same.

Thank you.



Dr. Agatha Wagutu

Ag. COD, PURE & APPLIED SCIENCES

APPENDIX 2: NACOSTI RESEARCH



REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **898582**

Date of Issue: **17/September/2025**

RESEARCH LICENSE

This is to Certify that Mr.. NICHOLAS MURIUKI MURIUKI of Kirinyaga University, has been licensed to conduct research as per the provision of the Science, Technology and Innovation Act, 2013 (Rev.2014) in on the topic: A CYBER SECURITY-BASED CONVOLUTION NEURAL NETWORK DETECTION MODEL FOR EMAIL PHISHING USING BI-LONG SHORT TERM MEMORY ALGORITHM for the period ending : 17/September/2026.

License No: **NACOSTI/P/25/4179878**

898582

Applicant Identification Number

Ag. Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.

See overleaf for conditions

APPENDIX 3: CODE SNIPPETS

```
# ------------------------

# IMPORTS

# ------------------------

import os

import pickle

import numpy as np

import pandas as pd
```

```
import matplotlib.pyplot as plt
from flask import Flask, request, render_template_string
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, roc_auc_score, confusion_matrix, roc_curve, auc
from gensim.downloader import load as gensim_load
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, LSTM, Dropout, Dense
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
# ------------------------
# CONFIG
# ------------------------
FILE_NAME = "phishing_legit_dataset_KD_10000.csv"
MODEL_PATH = "cnn_lstm_model.h5"
SCALER_PATH = "scaler.pkl"
EMBEDDING_DIM = 100
# ------------------------
# AUTO FILE SEARCH (FIXED)
# ------------------------
POSSIBLE_PATHS = [
    FILE_NAME,
    os.path.join(os.getcwd(), FILE_NAME),
    os.path.join(os.path.expanduser("~"), "Downloads", FILE_NAME),
```

```
    os.path.join(os.path.expanduser("~"), "Desktop", FILE_NAME),
]
CSV_PATH = None
for path in POSSIBLE_PATHS:
    if os.path.exists(path):
        CSV_PATH = path
        break
if CSV_PATH is None:
    print("Dataset not found!")
    print("Place file in Desktop / Downloads / project folder")
    raise SystemExit()
print("Dataset found at:", CSV_PATH)
# -------------------------
# LOAD DATA
# -------------------------
df = pd.read_csv(CSV_PATH)
df.columns = df.columns.str.strip().str.lower()
# detect columns
text_col = next(c for c in df.columns if "text" in c or "email" in c or "message" in c)
label_col = next(c for c in df.columns if "label" in c or "class" in c or "target" in c)
# clean duplicates
df[text_col] = df[text_col].astype(str).str.lower().str.strip()
df = df.drop_duplicates(subset=[text_col])
```

```
# normalize labels
df[label_col] = df[label_col].astype(str).str.lower().map({
    "phishing": 1,
    "spam": 1,
    "malicious": 1,
    "1": 1,
    "legit": 0,
    "legitimate": 0,
    "ham": 0,
    "0": 0
})
df = df.dropna(subset=[text_col, label_col])
print("Dataset shape:", df.shape)
# -------------------------
# SPLIT RAW TEXT FIRST (NO LEAKAGE)
# -------------------------
X_text = df[text_col].values
y = df[label_col].astype(int).values
X_train_text, X_test_text, y_train, y_test = train_test_split(
    X_text,
    y,
    test_size=0.3,
    random_state=42,
```

```
    stratify=y
)
X_train_text, X_val_text, y_train, y_val = train_test_split(
    X_train_text,
    y_train,
    test_size=0.2,
    random_state=42,
    stratify=y_train
)
# -------------------------
# LOAD EMBEDDINGS
# -------------------------
print("Loading Word Embeddings...")
w2v = gensim_load("glove-wiki-gigaword-100")
print("Loaded.")
def text_to_vec(text):
    words = str(text).split()
    vecs = [w2v[w] for w in words if w in w2v.key_to_index]
    return np.mean(vecs, axis=0) if vecs else np.zeros(EMBEDDING_DIM)
def vectorize(texts):
    return np.array([text_to_vec(t) for t in texts])
# -------------------------
# VECTORIZE AFTER SPLIT
```

```
# -------------------------

X_train = vectorize(X_train_text)

X_val = vectorize(X_val_text)

X_test = vectorize(X_test_text)

# -------------------------

# ADD NOISE (PREVENT MEMORIZATION)

# -------------------------

X_train = X_train + np.random.normal(0, 0.01, X_train.shape)

# -------------------------

# SCALE

# -------------------------

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_val = scaler.transform(X_val)

X_test = scaler.transform(X_test)

pickle.dump(scaler, open(SCALER_PATH, "wb"))

# -------------------------

# RESHAPE FOR CNN/LSTM

# -------------------------

timesteps = X_train.shape[1]

X_train = X_train.reshape(-1, timesteps, 1)

X_val = X_val.reshape(-1, timesteps, 1)

X_test = X_test.reshape(-1, timesteps, 1)
```

```
# ------------------------
# MODEL
# ------------------------
def build_model():
    model = Sequential([
        Conv1D(32, 3, activation="relu", input_shape=(timesteps, 1)),
        MaxPooling1D(2),
        LSTM(32),
        Dropout(0.5),
        Dense(32, activation="relu"),
        Dropout(0.5),
        Dense(1, activation="sigmoid")
    ])
    model.compile(
        optimizer="adam",
        loss="binary_crossentropy",
        metrics=["accuracy"]
    )
    return model
model = build_model()
# ------------------------
# TRAIN
# ------------------------
```

```
checkpoint = ModelCheckpoint(MODEL_PATH, save_best_only=True)
early = EarlyStopping(patience=5, restore_best_weights=True)
model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    epochs=5,
    batch_size=32,
    callbacks=[checkpoint, early],
    verbose=1
)
# -------------------------
# EVALUATION
# -------------------------
y_probs = model.predict(X_test).ravel()
y_pred = (y_probs > 0.5).astype(int)
print("\nFINAL RESULTS:")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("ROC-AUC:", roc_auc_score(y_test, y_probs))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
fpr, tpr, _ = roc_curve(y_test, y_probs)
roc_auc_value = auc(fpr, tpr)
plt.figure()
plt.plot(fpr, tpr, label=f"AUC = {roc_auc_value:.4f}")
```

```
plt.plot([0, 1], [0, 1], "--")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
print("ROC-AUC:", roc_auc_value)
# -------------------------
# Flask App
# -------------------------
app = Flask(__name__)
HTML_TEMPLATE = """
<!doctype html>
<html>
<head>
<title>Phishing Detector Explained</title>
<style>
body {font-family: Arial; background:#f5f5f5; text-align:center; padding:40px;}
.box {background:white; padding:20px; width:700px; margin:auto; border-radius:10px;}
textarea {width:100%; height:120px;}
button {padding:10px; background:#007BFF; color:white; border:none;}
.result {margin-top:20px; text-align:left; white-space:pre-line;}
</style>
```

```
</head>
<body>
<div class="box">
<h2>Phishing Detector (Explainable MODEL [CNN-LSTM)</h2>
<form method="POST" action="/predict">
<textarea name="text" placeholder="Paste email text here"></textarea><br><br>
<button type="submit">Analyze</button>
</form>
{% if result %}
<div class="result">{{ result }}</div>
{% endif %}
</div>
</body>
</html>
@app.route("/")
def home():
    return render_template_string(HTML_TEMPLATE)
@app.route("/predict", methods=["POST"])
def predict():
    text = request.form.get("text", "")
    # Feature extraction
    text_vec = text_to_avg_w2v(text, w2v)
    if numeric_cols:
```

```
        numeric_values = np.zeros(len(numeric_cols))
        full_feat = np.hstack([numeric_values, text_vec])
    else:
        full_feat = text_vec
    full_feat_scaled = scaler.transform(full_feat.reshape(1, -1))
    full_feat_cnn = full_feat_scaled.reshape(1, timesteps, 1)
    prob = float(model.predict(full_feat_cnn)[0][0])
    pred = 1 if prob > 0.5 else 0
    label = "Phishing" if pred == 1 else "Legitimate"
    explanation = explain_phishing(text)
    if pred == 1:
        reason = "Model detected phishing language patterns + suspicious structure."
    else:
        reason = "No strong phishing patterns detected."
    result_text = f"""
Prediction: {label}
Confidence: {prob:.2f}
Why this result:
{reason}
Text Explanation:
{explanation}
    return render_template_string(HTML_TEMPLATE, result=result_text)
# ------------------------
```

```
# Run
# ------------------------
if __name__ == "__main__":
    print("Server running at http://127.0.0.1:5000")
    app.run(debug=False)
```

APPENDIX 4: QUESTIONNAIRE

A Cyber Security-Based Convolution Neural Network Detection Model for Email Phishing Using Long Short Term Memory Algorithm Questionnaire

Dear Respondents,

My name is Muriithi Nicholas, and I am pursuing Doctor of Philosophy in Information Technology (PhD IT) at Kirinyaga University. In my doctoral research projects, I am conducting a study on the design and testing of a cybersecurity-based phishing email detection model that incorporates Convolutional Neural Networks (CNNs) and the Long Short-Term Memory (LSTM) algorithm.

This questionnaire is aimed at gathering knowledge and professional views about how the given model can identify phishing emails and valid emails to minimize the risks of cybersecurity. The questionnaire will also be used to confirm the effectiveness, reliability and performance of CNN and LSTM based model in the detection of phishing threats in the email communication system. The gathered responses through the participation of experts will help in testing the proposed phishing detection model especially when it comes to determining the capability of the model to examine email characteristics like emails headers, messages, subject lines, and URLs in order to correctly differentiate between phishing and genuine emails. Your response will thus be significant in the identification of whether the developed model will match the anticipated technical and

usability requirements in the cybersecurity applications.

You are hereby asked to mark (✓) next to the correct box ( ) next to the answer that will best describe your answer to each question presented in the questionnaire. The sincere and professional answers that you provide will significantly help in the validation and enhancement of the suggested phishing detection model.

All replies will be viewed with the strict confidentiality and the received information will be utilized only in the academic research.

It is with your cooperation and time that we appreciate you in making and contributing a significant contribution to this research study.

Section A: Demographic Information

1. What is your Gender * (Select one)

☐ Male

☐ Female

☐ Prefer not to say

2. What is your Age Group * (Select one)

☐ Below 20

☐ 21-30

☐ 31-40

☐ 41-50

☐ Above 50

3. What is your Education Level * (Select one)

☐ Diploma

☐ Bachelor's Degree

☐ Master's Degree

☐ PhD

☐ Other: _____

4. What is your Area of Expertise * (Select one)

☐ Cyber Security

☐ Computer Science

☐ Information Technology

☐ Data Science

☐ Other: _____

5. Experience in Email Security / Machine Learning * (Select one)

☐ Less than 1 Year

☐ 1-3 Years

☐ 4-6 Years

☐ Over 6 Years

6. Have you previously used or tested phishing detection systems? *

☐ Yes

☐ No

Section B: Technical Evaluation of the Phishing Detection Model

Objective: To evaluate whether the model achieves the expected performance using CNN and LSTM algorithms.

Scale: 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree

7. The CNN + LSTM model effectively distinguishes between phishing and legitimate emails.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

8. The model efficiently analyzes multiple email features (headers, content, subject, URLs).

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

9. The model architecture (CNN + LSTM) provides better accuracy than traditional machine learning models.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

10. The training data from the Kaggle Data repository are appropriate and reliable for phishing detection.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

11. The model achieves consistent accuracy and minimal overfitting after multiple epochs.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

12. The confusion matrix results show strong predictive power and reliability.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

13. The use of CNN for feature extraction and LSTM for sequence learning improves detection performance.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

14. The developed model demonstrates good generalization and low false alarm rate.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

15. The system can be applied in real-world email filtering and cybersecurity operations.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

Section C: Usability and System Interface Evaluation

Scale: 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree

16. The Flask API interface is easy to use and understand.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

17. The system provides responses quickly without significant delay.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

18. The system can be easily integrated into existing email platforms or servers.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

19. The instructions provided on the system interface are sufficient.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

20. The system enhances user confidence in identifying safe and unsafe emails.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

21. Overall, the system provides a satisfactory user experience.

1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐

Section D: Overall Perception and Recommendations

22. What do you find most effective about the CNN + LSTM phishing detection model?

23. What improvements would you recommend for better model performance or usability?