ARTIFICIAL INTELLIGENCE FOR

SOCIAL ENGINEERING DETECTION

NURSYUHADAH BINTI AHMAD SUDERMAN

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**BORANG PENGESAHAN STATUS LAPORAN**

JUDUL: [ARTIFICIAL INTELLIGENCE FOR SOCIAL ENGINEERING DETECTION]

SESI PENGAJIAN: [2024 / 2025]

Saya: \_\_\_\_[NURSYUHADAH BINTI AHMAD SUDERMAN]\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

mengaku membenarkan tesis Projek Sarjana Muda ini disimpan di Perpustakaan Universiti Teknikal Malaysia Melaka dengan syarat-syarat kegunaan seperti berikut:

1. Tesis dan projekadalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan unituk tujuan pengajian sahaja.
3. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. \* Sila tandakan (✓)

|  |  |  |
| --- | --- | --- |
| \_\_\_\_\_\_\_\_\_\_ | SULIT | (Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972) |
| \_\_\_\_\_\_\_\_\_\_ | TERHAD | (Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi / badan di mana penyelidikan dijalankan) |
| \_\_\_\_✓\_\_\_\_ | TIDAK TERHAD |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| (TANDATANGAN PELAJAR) |  | (TANDATANGAN PENYELIA) |
| Alamat tetap: E-7-9 P/A DESA REJANG PERSIARAN REJANG, TAMAN SETAPAK JAYA 53300, WILAYAH PERSEKUTUAN KUALA LUMPUR |  | DR ZAHEERA BINTI ZAINAL ABIDIN |
|  |  | Nama Penyelia |
| Tarikh: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Tarikh: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

CATATAN: \* Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa.



ARTIFICIAL INTELLIGENCE FOR SOCIAL ENGINEERING DETECTION

NURSYUHADAH BINTI AHMAD SUDERMAN

This report is submitted in partial fulfillment of the requirements for the

Bachelor of [Computer Science (Computer Security)] with Honours.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2025

# DECLARATION

I hereby declare that this project report entitled

[ARTIFICIAL INTELLIGENCE FOR SOCIAL ENGINEERING DETECTION

is written by me and is my own effort and that no part has been plagiarized

without citations.

STUDENT : NURSYUHADAH BINTI AHMAD SUDERMAN Date : \_\_\_\_\_\_\_\_

I hereby declare that I have read this project report and found

this project report is sufficient in term of the scope and quality for the award of

Bachelor of [Computer Science (Computer Security)] with Honours.

SUPERVISOR : DR ZAHEERA BINTI ZAINAL ABIDIN Date : \_\_\_\_\_\_\_\_

# DEDICATION

This work is dedicated to Allah Almighty, my creator, pillar, and source of wisdom, knowledge, and insight. Throughout this journey, He has been my source of strength, and I have only been able to fly on His wings.

To my beloved parents, Ahmad Suderman Bin Mohd Yaacob and Norayzuazlin Binti Mohd Ayob whom I wholeheartedly dedicate this study to. Thank you for being my source of inspiration and for your unwavering support and trust. To my brothers, whom I hold very dear to my heart.

To my supervisor, Dr Zaheera binti Zainal Abidin, for supporting and believing in me to accomplish this project. Finally, I want to thank my friends for being there for me during my bachelor's degree, sharing words of advice and support to help me accomplish this study.

# ACKNOWLEDGEMENTS

First and foremost, First and foremost, I would like to praise and thank Allah the Almighty, the Most Gracious, and the Most Merciful for His blessing given to me in completing this project. Peace and salutations to the Prophet Muhammad S.A.W, who has led all humanity from darkness to light.

I would like to express my heartfelt appreciation to Dr. Zaheera binti Zainal Abidin for her assistance in completing this assignment successfully. Her insightful thoughts have provided various inspirations and enlightenment for this project.

I'd also want to thank my dear parents and sisters for their encouragement and support during this effort. Their help and concern had given me confidence while I worked on this project. Finally, I am grateful to my friends who are ready to share their wisdom and have directly or indirectly guided me during this project.

# ABSTRACT

Social engineering attacks pose a significant and growing threat in the cybersecurity landscape, exploiting human psychology rather than technical vulnerabilities to manipulate individuals into compromising sensitive information. Among various social engineering techniques, phishing has emerged as the most prevalent and damaging, driven by its scalability, low execution cost, and evolving tactics such as spear phishing and smishing. As traditional rule-based detection methods struggle to keep up with these dynamic threats, Artificial Intelligence (AI) offers a robust and adaptive solution. This study explores AI-based approaches—specifically Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL)—for the detection of phishing attacks. It provides a comprehensive literature review, analysis of current research trends (2020–2024), and detailed examination of relevant datasets and algorithms including Support Vector Machines (SVM), Long Short-Term Memory (LSTM), and Deep Q-Networks (DQN). The research highlights the advantages of AI in automating threat detection, enhancing accuracy, and enabling systems to adapt to new attack vectors. By focusing on phishing as a primary social engineering threat, this work aims to contribute to the development of intelligent, scalable, and real-time defenses against increasingly sophisticated cyber threats.

# ABSTRAK

Serangan kejuruteraan sosial merupakan ancaman yang semakin ketara dalam landskap keselamatan siber, di mana ia mengeksploitasi psikologi manusia dan bukannya kelemahan teknikal untuk memanipulasi individu agar mendedahkan maklumat sensitif. Dalam pelbagai teknik kejuruteraan sosial, serangan *phishing* telah muncul sebagai bentuk yang paling meluas dan merosakkan, didorong oleh keupayaannya untuk berkembang secara besar-besaran, kos pelaksanaan yang rendah, serta taktik yang sentiasa berubah seperti *spear phishing* dan *smishing*. Oleh kerana kaedah pengesanan berasaskan peraturan tradisional semakin sukar menandingi ancaman yang dinamik ini, Kecerdasan Buatan (AI) menawarkan penyelesaian yang kukuh dan adaptif. Kajian ini meneroka pendekatan berasaskan AI—khususnya Pembelajaran Mesin (ML), Pembelajaran Mendalam (DL), dan Pembelajaran Penguatan (RL)—dalam pengesanan serangan *phishing*. Ia merangkumi ulasan literatur yang komprehensif, analisis *trend* penyelidikan semasa, serta pemeriksaan terperinci terhadap set data dan algoritma yang relevan termasuk *Support Vector Machines* (SVM), *Long Short-Term Memory* (LSTM), dan *Deep Q-Networks* (DQN). Kajian ini menekankan kelebihan AI dalam mengautomasikan pengesanan ancaman, meningkatkan ketepatan, dan membolehkan sistem menyesuaikan diri dengan vektor serangan baharu. Dengan memberi tumpuan kepada *phishing* sebagai ancaman utama kejuruteraan sosial, kajian ini bertujuan menyumbang kepada pembangunan sistem pertahanan yang pintar, berskala, dan masa nyata terhadap ancaman siber yang semakin canggih.

# TABLE OF CONTENTS

[DECLARATION ii](#_Toc196477482)

[DEDICATION iii](#_Toc196477483)

[ACKNOWLEDGEMENTS iv](#_Toc196477484)

[ABSTRACT v](#_Toc196477485)

[ABSTRAK vi](#_Toc196477486)

[TABLE OF CONTENTS vii](#_Toc196477487)

[list of tables x](#_Toc196477488)

[list of figures xi](#_Toc196477489)

[List of Abbreviations xiii](#_Toc196477490)

[List of ATTACHMENTS xiv](#_Toc196477491)

[Chapter 1: INTRODUCTION 1](#_Toc196477492)

[1.1 Introduction 1](#_Toc196477493)

[1.2 Problem Statement 2](#_Toc196477494)

[1.3 Project Question 2](#_Toc196477495)

[1.4 Objective 3](#_Toc196477496)

[1.5 Project Scope 4](#_Toc196477497)

[1.6 Project Contribution 4](#_Toc196477498)

[1.7 Thesis Organization 4](#_Toc196477499)

[1.8 Conclusion 4](#_Toc196477500)

[Chapter 2: literature review 5](#_Toc196477501)

[2.1 Introduction 5](#_Toc196477502)

[2.2 Related Work 7](#_Toc196477503)

[2.2.1 Phishing 8](#_Toc196477504)

[2.2.2 Machine Language Approaches 9](#_Toc196477505)

[2.2.3 Deep Learning Approaches 17](#_Toc196477506)

[2.2.4 Reinforcement Learning Approaches 22](#_Toc196477507)

[2.3 Analysis 23](#_Toc196477508)

[2.4 Critical Review ( kene ada features, math equation, justification) 23](#_Toc196477509)

[2.5 Project Solution (kene tahu algorithm, study algorithm) 23](#_Toc196477510)

[2.6 Summary 23](#_Toc196477511)

[Chapter 3: METHODOLOGY 24](#_Toc196477512)

[3.1 Introduction 24](#_Toc196477513)

[3.2 Methodology 24](#_Toc196477514)

[3.2.1 Requirement Analysis 24](#_Toc196477515)

[3.2.2 System Design 25](#_Toc196477516)

[3.2.3 Implementation 25](#_Toc196477517)

[3.2.4 Testing 25](#_Toc196477518)

[3.2.5 Deployment 25](#_Toc196477519)

[3.2.6 Maintenance 25](#_Toc196477520)

[3.3 Project Schedule and Milestones 27](#_Toc196477521)

[3.3.1 Project Milestones 27](#_Toc196477522)

[3.3.2 Project Gantt Chart 28](#_Toc196477523)

[3.4 Summary 29](#_Toc196477524)

[Chapter 4: ANALYSIS AND DESIGN 30](#_Toc196477525)

[4.1 Introduction 30](#_Toc196477526)

[4.2 Summary 30](#_Toc196477527)

[Chapter 5: IMPLEMENTATION 31](#_Toc196477528)

[5.1 Introduction 31](#_Toc196477529)

[5.2 Summary 31](#_Toc196477530)

[Chapter 6: discussion 31](#_Toc196477531)

[6.1 Introduction 31](#_Toc196477532)

[6.2 Result and Analysis 31](#_Toc196477533)

[6.3 Summary 31](#_Toc196477534)

[Chapter 7: CONCLUSION 32](#_Toc196477535)

[7.1 Introduction 32](#_Toc196477536)

[7.2 Project Summarization 32](#_Toc196477537)

[7.3 Project Contribution 32](#_Toc196477538)

[7.4 Project Limitation 32](#_Toc196477539)

[7.5 Future Works 32](#_Toc196477540)

[7.6 Summary 32](#_Toc196477541)

[references 33](#_Toc196477542)

# list of tables

|  |  |
| --- | --- |
|  | **PAGE** |

[Table 1.2‑1 Problem Statement 2](#_Toc112842306)

[Table 1.3‑1 Project Question 3](#_Toc112842307)

[Table 1.4‑1 Project Objective 4](#_Toc112842308)

[Table 1.6‑1 Project Contribution 5](#_Toc112842309)

[Table 2.4‑1: Summary of Research Papers Reviewed 22](#_Toc112842310)

[Table 3.2‑1: Hardware Requirement 28](#_Toc112842311)

[Table 3.2‑2: Software Requirement 29](#_Toc112842312)

[Table 3.2‑3: Example of Comparison Table generated after analysis of data is finished 31](#_Toc112842313)

[Table 4.4‑1: Possible Scenarios 40](#_Toc112842314)

[Table 5.4‑1 below shows the value of bandwidth usage, latency and throughput respective to the three protocols studied. 56](#_Toc112842315)

[Table 5.4‑1: Comparison Table from Collected Data 56](#_Toc112842316)

[Table 5.5‑1: Implementation Status 57](#_Toc112842317)

# list of figures

|  |  |
| --- | --- |
|  | **PAGE** |

Figure 1: Literature Review Structure Diagram 8

Figure 2: Action varieties in Social Engineering incidents 9

Figure 3: CoAP Protocol Layers (payatu.com) 10

Figure 4: Confirmable Message (payatu.com) 11

Figure 5: Non-confirmable message (payatu.com) 11

Figure 6: Publish/Subscribe model in MQTT (Martí et al., 2019) 13

Figure 7: QoS0 (Sueda et al., 2019) 14

Figure 8: QoS1 (Sueda et al., 2019) 15

Figure 9: QoS2 (Sueda et al., 2019) 15

Figure 10: AMQP Architecture (Anusha, et al., 2017) 16

Figure 11: Sensors used to collect temperature, humidity and light (Corak et al., 2018) 18

Figure 12: Diagram of testbed’s processing unit (Corak et al., 2018) 18

Figure 13: Topology for (a) Experiment I and (b) Experiment II (Moraes et al., 2019) 19

Figure 14: Diagram of hardware used for the testbed (Pohl et al., 2018) 20

Figure 15: Experimental setup (Mijovic et al., 2016) 21

Figure 16 Waterfall Model 28

Figure 17: Data Processing and Obtaining Result 30

Figure 18: Example of Graph that compares three protocols based on Bandwidth Utilisation 31

Figure 19: Network System Architecture 36

Figure 20: Logical Design 38

Figure 21: Physical Design 39

Figure 22: Scenario 1: CoAP 40

Figure 23: Scenario 2: AMQP 40

Figure 24: Scenario 3: MQTT 41

Figure 25: CoAP server running at Server 44

Figure 26: CoAP client running at PC1 44

Figure 27: CoAP Client sending GET message 45

Figure 28: Server received CoAP messages from PC1 45

Figure 29: Wireshark Packet Capture (CoAP) 46

Figure 30: RabbitMQ installed in Server browser 46

Figure 31: Python code for sending message 47

Figure 32: Python code for receiving message 48

Figure 33: RabbitMQ Management Interface 48

Figure 34: Wireshark Packet Capture (AMQP) 49

Figure 35: Moisquitto loaded and running 50

Figure 36: Connection details 50

Figure 37: PC1 MQTTlens interface 51

Figure 38: PC2 MQTTlens interface 51

Figure 39: PC2 received 100 messages from PC1 52

Figure 40: Wireshark Packet Capture (MQTT) 52

Figure 41: Protocol Hierarchy for CoAP 53

Figure 42: Protocol Hierarchy for AMQP 53

Figure 43: Protocol Hierarchy for MQTT 54

Figure 44: Packet Length for CoAP 54

Figure 45: Packet Length for AMQP 55

Figure 46: Packet Length for MQTT 55

Figure 47: Conversations Statistics for CoAP 56

Figure 48: Conversations Statistics for AMQP 56

Figure 49: Conversations Statistics for MQTT 56

Figure 50: Bandwidth Usage Graph 60

Figure 51: Latency Graph 61

Figure 52: Throughput Graph 61

Figure 53: Average Packet Size, B (CoAP) from Wireshark 62

Figure 54: Average Packet Size, B (MQTT) from Wireshark 62

# List of Abbreviations

|  |  |  |
| --- | --- | --- |
| **IoT** | **-** | **Internet of Things** |
| **CoAP** | **-** | **Constrained Application Protocol** |
| **AMQP** | **-** | **Advanced Message Queuing Protocol** |
| **MQTT** | **-** | **Message Queue Telemetry Transport** |
| **TCP** | **-** | **Transmission Control Protocol** |
| **UDP** | **-** | **User Datagram Protocol** |

# List of ATTACHMENTS

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | **PAGE** |
|  | | |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# INTRODUCTION

## Introduction

Social engineering attacks are among the most persistent and evolving cybersecurity threats, leveraging psychological manipulation to deceive individuals into disclosing confidential information or compromising systems. Techniques such as phishing, baiting, and pretexting exploit trust, urgency, or curiosity, making them difficult to detect using traditional, rule-based security systems. Phishing stands out as a dominant threat due to its scalability, ease of execution, and the increasing sophistication of its tactics.

This study explores the role of Artificial Intelligence (AI) in detecting social engineering attacks, with a primary focus on phishing. The research involves a comprehensive analysis of existing AI approaches—specifically Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL)—to evaluate their effectiveness, challenges, and potential in improving detection accuracy. Rather than developing a prototype, this study critically reviews current methods, datasets, and models to identify strengths, limitations, and areas for improvement in AI-based social engineering detection systems.

## Problem Statement

The problem that has been identified is summarized in Table 1-1 below:

Table 1.2‑1 Problem Statement

|  |  |
| --- | --- |
| **PS** | **Problem Statement** |
| **PS1** | Existing AI models struggle to detect the constantly changing nature of social engineering attacks, resulting in lower detection accuracy. |

**PS1: Existing AI models struggle to detect the constantly changing nature of social engineering attacks, resulting in lower detection accuracy.**

Existing AI models face challenges in accurately detecting social engineering attacks due to the dynamic and evolving tactics used by attackers. Machine learning has gradually introduced a huge basket of “AI capabilities” that can be harnessed for social engineering and phishing attacks (Schmitt & Flechais, 2024). This constant change reduces the effectiveness of traditional detection approaches, leading to lower overall detection accuracy.

## Project Question

Three Project Question (PQ) is constructed based on the problem statement that needs to be answered in this project. The summary of project question is shown in Table 1-2.

Table 1.3‑1 Project Question

|  |  |  |
| --- | --- | --- |
| **PS** | **PQ** | **Project Question** |
| PS1 | PQ1 | |  | | --- | |  |  |  | | --- | | What are the factors causing low detection accuracy in existing AI models when identifying evolving social engineering attacks? | |
| PQ2 | How can AI models be improved to adapt to the evolving tactics of social engineering attacks? |
| PQ3 | Which AI techniques or features are most effective in enhancing the detection accuracy against dynamic social engineering attacks? |

**PQ1: What are the factors causing low detection accuracy in existing AI models when identifying evolving social engineering attacks?**

Investigate and analyze the main factors that contribute to low detection accuracy of AI models against evolving social engineering tactics.

**PQ2: How can AI models be improved to adapt to the evolving tactics of social engineering attacks?**

Propose potential improvements and techniques to enhance AI models' adaptability to new and dynamic social engineering strategies.

**PQ3: Which AI techniques or features are most effective in enhancing the detection accuracy against dynamic social engineering attacks?**

Identify and evaluate the most effective AI techniques or feature combinations that improve detection accuracy against constantly evolving social engineering attacks.

## Objective

Based on the project questions formulated in previous section, appropriate project objectives (PO) are developed. The Project Objective (PO) is summarized in Table 1-3.

Table 1.4‑1 Project Objective

|  |  |  |  |
| --- | --- | --- | --- |
| **PS** | **PQ** | **PO** | **Project Objective** |
| PS1 | PQ1 | PO1 | To investigate existing AI models used for detecting social engineering attacks. |
| PQ2 | PO2 | To assess the effectiveness of linguistic, behavioral, and contextual features in AI-based detection. |
| PQ3 | PO3 | To identify key challenges and limitations of AI in detecting social engineering attacks. |

**PO1:** **To investigate existing AI models used for detecting social engineering attacks.Project Scope.**

Conduct a detailed analysis of current AI models applied in social engineering detection, focusing on their capabilities, detection methods, and areas where they fail to adapt to evolving attack tactics.

**PO2: To assess the effectiveness of linguistic, behavioral, and contextual features in AI-based detection.**

Evaluate how linguistic, behavioral, and contextual features contribute to the performance of AI models in detecting social engineering attacks, identifying which features are most influential.

**PO3: To identify key challenges and limitations of AI in detecting social engineering attacks.**

Analyze and summarize major challenges faced by AI systems, including model adaptability, evolving attack patterns, and data privacy concerns, impacting the effectiveness of social engineering detection.

## Project Scope

The project’s scopes are focused on the following aspects:

1. The AI tools considered for social engineering attack detection are Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL).
2. The performance of the AI model will be evaluated based on accuracy, precision, recall, and F1-score.
3. The platform used for training and testing the AI models is WEKA and KNIME

## Project Contribution

## Thesis Organization

This report is divided into six chapters, namely Chapter 1: Introduction, Chapter 2: Literature Review, Chapter 3: Methodology, Chapter 4: Design, Chapter 5: Implementation, Chapter 6: Discussion and Chapter 7: Conclusion.

**Chapter 1: Introduction**

This chapter discusses the project's introduction, background, research problem, research questions, objectives, scope, significance of the project, and report organization. It sets the stage for the problem addressed and outlines the goals of the project in the context of AI-based social engineering attack detection.

**Chapter 2: Literature Review**

This chapter provides a review of relevant literature, discussing previous research on social engineering attacks, existing detection methods, and AI techniques such as Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL). It also identifies gaps in the current literature and frames the research questions and objectives.

**Chapter 3: Methodology**

This chapter outlines the research methodology adopted for the project, including the datasets used, AI techniques implemented (ML, DL, RL), and the performance evaluation metrics. It provides a step-by-step explanation of the model development, data preprocessing, feature extraction, and model training processes.

**Chapter 4: Analysis and Design**

**Chapter 5: Implementation**

**Chapter 6: Discussion**

**Chapter 7: Project Conclusion**

## Summary

# literature review

## Introduction

**Social engineering** is a psychological manipulation technique where cybercriminals exploit human trust, fear, or urgency to deceive individuals into performing actions that compromise security, such as disclosing confidential information or enabling system access. Instead of relying on traditional hacking tools to exploit software or hardware vulnerabilities, social engineering exploits human behavior, making it one of the most dangerous and successful forms of attack in the modern threat landscape. Techniques include phishing, baiting, pretexting, vishing, and quid pro quo.

Among these techniques, **phishing remains the most widespread and damaging form of social engineering attack**, and its prevalence has been supported by numerous global studies and industry analyses. Research shows that phishing is consistently ranked as the most common social engineering threat due to its effectiveness, ease of execution, and adaptability across communication channels (Alzahrani et al., 2021; Basit et al., 2022). Attackers exploit digital platforms, particularly email, to impersonate legitimate institutions and deceive users into providing sensitive information.

What sets phishing apart from other social engineering methods is its ability to scale rapidly and target both individuals and organizations with minimal resources. Unlike baiting or pretexting, phishing campaigns can be automated and distributed widely using email bots and phishing kits available on underground forums. These features make phishing a recurring top threat in annual cybersecurity threat reports by multiple academic and professional organizations.

The continued evolution of phishing tactics, from generic spam to personalized spear phishing and smishing (SMS phishing), has contributed to its resilience and growth. This explains why, compared to other forms of social engineering attacks, phishing is the focus of this research. It provides not only a well-documented attack vector but also a diverse set of datasets and practical applications for testing AI-based detection models.

As the complexity and frequency of phishing attacks grow, the need for intelligent, adaptive defense systems becomes critical. Traditional security measures often fall short, especially against zero-day phishing attacks or personalized spear phishing. This has led to a strong shift toward **Artificial Intelligence (AI)**, specifically **Machine Learning (ML)**, **Deep Learning (DL)**, and **Reinforcement Learning (RL)**, as a robust solution for phishing detection. AI-based models can process large volumes of data, identify hidden patterns, and adapt over time, enabling more accurate and timely threat detection.

Given phishing's prominence and evolving nature, this study focuses exclusively on AI-based phishing detection. Compared to other forms of social engineering, phishing offers both a greater volume of research data and clearer use cases for AI implementation, making it an ideal focal point for academic and practical analysis. The subsequent sections provide a deep dive into AI techniques, available software tools, datasets, and recent research that contribute to building effective, intelligent phishing detection systems.



Figure 1: Literature Review Structure Diagram

## Related Work

Recent advancements between 2020 and 2024 have demonstrated the growing impact of AI in combating phishing threats. Researchers have developed increasingly sophisticated models leveraging ML, DL, and RL to classify, detect, and respond to phishing activities in real time.

Rahman et al. (2022) introduced a hybrid model combining Decision Trees and anomaly detection, achieving improved phishing email detection rates in enterprise environments. Their work highlighted the effectiveness of ensemble methods in capturing varied attack patterns. Aljohani and Hossain (2021) employed the BERT transformer model to detect phishing content by analyzing contextual semantics within emails. Their results showed significant accuracy improvements over traditional ML models.

Zhao et al. (2020) used Long Short-Term Memory (LSTM) networks to detect phishing attempts in mobile text messages. LSTM’s ability to process sequential information proved critical in identifying suspicious messages that mimic legitimate communications. Nasir et al. (2023) proposed a CNN-based phishing detection system that integrated user behavior logs with content analysis. The combination of behavioral and textual features enhanced detection performance, particularly for spear-phishing cases.

Asker and Essa (2024) emphasized the potential of unsupervised models, such as Autoencoders, to identify phishing emails without labelled data. Their findings suggest these models are well-suited for environments where phishing strategies evolve rapidly and labelled examples are scarce. Zhang et al. (2023) explored the use of Reinforcement Learning to create adaptive phishing detection systems. Their RL-based framework optimized decision-making policies in real-time, improving resilience against evolving attack vectors while minimizing false positives.

These studies demonstrate a transition from static rule-based systems to adaptive, intelligent models that integrate language understanding, behavior analysis, and real-time learning. They underscore the importance of hybrid approaches and model interpretability in building effective phishing detection solutions.

### Phishing

Phishing is a cyber attack technique in which malicious actors impersonate trustworthy entities to deceive victims into revealing sensitive information, such as usernames, passwords, credit card details, or login credentials. These attacks often take the form of emails, websites, or messages that closely resemble legitimate communication from known organizations. The purpose of phishing is typically to commit fraud, gain unauthorized access to systems, or install malware.

Phishing continues to be the most prevalent type of social engineering attack globally. Its popularity among attackers stems from its high success rate, scalability, and ability to exploit human behavior rather than technical vulnerabilities. According to the 2023 Verizon Data Breach Investigations Report, phishing accounted for more than one-third of all social engineering-related incidents. The Anti-Phishing Working Group (APWG) also reported over 1.2 million phishing attacks in a single quarter, indicating a steady and alarming upward trend. These attacks have evolved from generic bulk emails to more targeted approaches such as spear phishing, business email compromise (BEC), and smishing (SMS phishing), which are increasingly difficult to detect. Figure below shows that Pretexting is now more prevalent than Phishing in Social Engineering incidents. However, when we look at confirmed breaches, Phishing is still on top (Verizon, 2023).



**Figure 2: Action varieties in Social Engineering incidents (Verizon, 2023)**

Due to the sophistication and frequency of phishing, AI-based techniques have become central to its detection. Machine Learning approaches such as Support Vector Machine (SVM) and Random Forest are commonly used due to their strong classification capabilities on structured datasets. These models analyze features such as email headers, URL characteristics, and content patterns., models like Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) have proven effective in understanding the semantic structure of emails, enabling the identification of sophisticated phishing attempts that evade traditional filters.

Recently, Reinforcement Learning (RL) has emerged as a powerful approach for dynamic and adaptive phishing detection. Unlike ML and DL, which learn from static datasets, RL enables an agent to learn optimal policies through interactions with an environment, receiving feedback in the form of rewards or penalties. Techniques such as Q-learning, Deep Q-Networks (DQN), and Policy Gradient Methods are used to develop adaptive systems that can fine-tune detection thresholds based on evolving user behavior and attack strategies. These RL-based methods are especially useful in real-time systems, where phishing tactics change frequently and require continuous learning to maintain detection accuracy.

### Machine Language Approaches

**Machine Learning (ML)** is a core subset of Artificial Intelligence (AI) that focuses on creating algorithms and models that enable computers to automatically learn and improve from experience without being explicitly programmed for specific tasks. At its essence, ML empowers systems to identify patterns, make predictions, and adapt to new data through iterative learning processes. This adaptability is achieved by training models on vast amounts of labelled or unlabelled data and optimizing performance based on measurable outcomes (GeeksforGeeks, 2023).

Modern ML approaches are categorized into three main types: supervised learning, where models learn from labelled data; unsupervised learning, where models uncover hidden structures in unlabelled data; and reinforcement learning, where agents learn optimal behaviors by interacting with environments and receiving feedback in the form of rewards or penalties (Kelleher, 2020). ML is widely applied across domains such as cybersecurity, finance, healthcare, and e-commerce due to its ability to process and learn from high-dimensional, complex datasets.

In the context of cybersecurity, ML plays a pivotal role in the detection and mitigation of threats like phishing attacks, malware, and network intrusions. Techniques such as Support Vector Machines (SVM), Random Forest, and Naïve Bayes have demonstrated high effectiveness in classifying malicious content by learning from historical attack patterns and extracting relevant features from structured or textual data (Alzahrani et al., 2021; Kaur & Arora, 2023).

#### Support Vector Machines (SVM)

Support Vector Machine (SVM) model as a hyper plane in which several classes are represented. So that we may reduce the amount of mistakes we make, SVM will create the hyper plane in an iterative fashion. SVM's purpose is to classify datasets such that a maximum marginal hyper plane may be discovered (Jain & Gupta, 2023).

The high accuracy of SVMs in phishing detection has been demonstrated in various studies. For instance, research by Gupta et al. (2023) highlights that SVMs, when properly tuned, can achieve remarkable detection rates and low false positive rates, making them a reliable choice for cybersecurity applications. However, the effectiveness of SVMs comes with certain challenges. The considerable parameter adjustment necessary to maximize the model's performance is one of its primary drawbacks. It can take some time to carefully choose parameters like the kernel parameters and the penalty parameter (C) to balance the choices between variance and bias.

The computational expense associated with Support Vector Machines (SVMs), particularly when using non-linear kernels, can be significant. Training an SVM model involves solving a complex optimization problem, which can be computationally intensive, especially when dealing with large datasets typically encountered in phishing detection systems. This computational burden can limit the practicality of SVMs in real-time phishing detection, where fast processing is crucial to ensure timely responses and effective protection. Despite these challenges, the high precision and robustness of SVMs in identifying complex patterns in data make them a valuable asset in phishing detection frameworks.

To mitigate these limitations, hybrid approaches that combine SVMs with other machine learning techniques have been explored. These methods aim to capitalize on the strengths of SVMs while addressing their computational drawbacks. For example, integrating SVMs with feature selection techniques can reduce data dimensionality, lessening the computational load and improving real-time applicability. As cyber threats continue to evolve, the role of SVMs in phishing detection remains critical, providing a balance of high accuracy and adaptability necessary for effective cybersecurity defences.



Figure 3: SVM algorithm (Jain & Gupta, 2023)

##### Attributes

To ensure that examples from different classes are as far apart as feasible, the SVM model represents examples as points in space. After that, fresh instances are shown within this identical area and categorized according to their placement along the margin (Patil et al., 2017). The following is a list of the attributes utilized for SVM classification:

|  |  |  |
| --- | --- | --- |
| Sr No | Features | Significance |
| 1 | Having IP Addr | If a domain name contains an IP address, the website is likely to be phishing. |
| 2 | URL\_Length | Legitimate URLs are typically around 75 characters long. URLs longer than 75 characters are likely to be phishing sites. |
| 3 | Shortening Service | Link softeners are used to deceive people. |
| 4 | Having\_At\_Symbol | Websites that contain an @ symbol are often considered suspicious. |
| 5 | Double slash redirecting | If a URL includes '/1', it may be classified as a phishing website. |
| 6 | Having Sub Domain | Usually, legitimate websites have a domain structure that has two levels. Since they could have multiple domains within a single domain, websites with more than three dots are frequently phishing sites. |
| 7 | URL of Anchor | The link tag on legitimate websites points to the same domain as the source code. Phishing websites, on the other hand, frequently have links to other domains. |
| 8 | Links in tags | Tag links could take you to fraudulent websites. |
| 9 | Abnormal URL | The primary identifier of legitimate websites is found in the URL; this element was taken from the Who is Database. |
| 10 | Age of domain | Websites that are older than six months are considered Phishy; those that are older than this are legitimate. Websites that are older than six months are considered Phishy; those that are older than this are legitimate. |
| 11 | Page Rank | Phishing websites usually don't have many links leading to them, so they have a poor page rank. |
| 12 | Links Pointing to page | Phishing websites often include links to zip files that automatically download malware. |

**Table 1: SVM Algorithm Attributes**

#### Random Forest

Random Forest is a well-known machine learning method that performs excellently when dealing with regression and classification problems. It creates a lot of decision trees during training, outputs the mode of classes for classification tasks or the mean prediction for regression tasks and increases accuracy and generalization ability by voting or averaging over all trees. Because of the ensemble technique, the model resists overfitting. Because Random Forest can handle big datasets with high dimensionality and is flexible and effective, it is widely employed. It is also resistant to noise and outliers and provides insights into feature importance. Visualization techniques can aid in model evaluation and decision-making (Gunjan & Prasad, 2024).

##### Random Forest Technique

Random Forest (RF) is a powerful ensemble learning algorithm that has shown strong performance in phishing detection due to its ability to handle complex patterns and noisy data. It operates by building multiple decision trees using different random subsets of the training dataset—a process known as bootstrapping. Each tree is trained independently, which introduces diversity and reduces the risk of overfitting. This is especially beneficial in phishing detection, where malicious patterns can be subtle and varied across different instances. Once the forest is constructed, each decision tree contributes to the prediction process. In classification tasks such as distinguishing between phishing and legitimate activities, each tree casts a "vote" for a class label. The final decision is based on majority voting, where the class with the most votes becomes the output. This collective decision-making mechanism improves both accuracy and stability, making Random Forest well-suited for phishing detection systems that require consistent performance under diverse threat scenarios. Refer to the figure below, which illustrates how multiple decision trees independently classify an instance and then combine their outputs through majority voting to determine the final class.



**Figure 4: Random Forest Algorithm**

### Deep Learning Approaches

**Deep Learning (DL)** is a specialized subfield of Machine Learning (ML) that utilizes artificial neural networks with multiple layers—referred to as **deep neural networks**—to automatically learn complex patterns from large volumes of data. Inspired by the structure and function of the human brain, DL systems can model intricate relationships and hierarchical representations in data, enabling them to outperform traditional ML algorithms in tasks involving high-dimensional, unstructured inputs such as images, audio, and text.

In the context of cybersecurity, and particularly phishing detection, deep learning techniques offer significant advantages. Unlike conventional ML models that rely heavily on manually engineered features, DL models can perform **automatic feature extraction**, allowing them to learn hidden patterns in raw email content, URLs, or user interaction logs. Common DL models applied in phishing detection include:

* **Long Short-Term Memory (LSTM)**: A type of Recurrent Neural Network (RNN) that is effective in capturing sequential dependencies in phishing emails or SMS messages.
* **Convolutional Neural Networks (CNNs)**: Originally designed for image recognition, CNNs are also used in phishing URL and text classification due to their ability to detect spatial and contextual patterns.
* **Transformer models (e.g., BERT)**: These models understand the contextual meaning of words in sentences, making them highly effective in detecting sophisticated and linguistically deceptive phishing emails.

The adaptability and high accuracy of DL models make them increasingly vital for building robust, intelligent, and real-time phishing detection systems.

#### Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to effectively capture long-term dependencies in sequential data. In the context of phishing website detection, LSTM plays a crucial role in modeling temporal patterns and contextual features that may not be apparent in static data. As described by Elberri et al. (2024), LSTM was integrated with a Convolutional Neural Network (CNN) in a hybrid deep learning architecture aimed at improving phishing detection accuracy. While CNN excels at extracting spatial features from grayscale images generated from URL and web content data, LSTM contributes by analyzing the sequential behavior and dependencies within the data. This combination allows the model to better understand complex relationships, such as those found in URL structures and embedded script behavior typical of phishing attacks. The study demonstrated that the CNN-LSTM hybrid model outperformed standalone CNN and LSTM models in terms of accuracy, sensitivity, and precision, highlighting LSTM’s effectiveness in enhancing the classification of phishing and legitimate websites.

##### Structure of an LSTM neural network cell

LSTM is a variant of the RNN deep learning architecture designed specifically for tasks like time series analysis and classification. LSTM effectively uses a gating mechanism to deal with vanishing gradient problems in the training process.The LSTM memory cell has four gates named forgetting f, input gate i, control gate c, and output gate o. The fundamental configuration of the LSTM cell is presented in Figure 4 and it consists of the output of the previous memory cell Ct-1. (Elberri et al., 2024)

This neural network uses components such as the input signal at each time step Xt, the current memory cell Ct output, the previously hidden unit Ht − 1, and the currently hidden unit Ht. The forget gate determines the way in which the contribution from the previous time step is incorporated, resulting in a value ranging from zero to one for each datapoint in Ct-1. The input gate regulates the amount of input that is stored in the memory cell from the current time step. Meanwhile, the control gate updates the memory cell contents from Ct-1 to Ct. The output gate dictates the extent to which the internal state influences the external state at the current time step. The symbol ⊗ represents the element-wise multiplication of vector elements, while ⊕ signifies the summation of vector along with the application of the σ (sigma) function. To formulate the LSTM artificial neural network. (Elberri et al., 2024)

Diagram of a diagram of a cell state

AI-generated content may be incorrect.

**Figure 5: Structure of an LSTM neural network cell (Elberri et al., 2024)**

A math equations with black text

AI-generated content may be incorrect.

**Figure 6: Equations of LSTM (Elberri et al., 2024)**

#### Bidirectional Encoder Representations from Transformers (BERT)

Bidirectional Encoder Representations from Transformers (BERT) is a powerful deep learning model that has shown great effectiveness in detecting phishing attacks, particularly those involving social engineering through text-based communication. Phishing often relies on carefully crafted language to manipulate users into revealing sensitive information or clicking malicious links. BERT’s unique bidirectional architecture allows it to analyze both the left and right context of each word in a sentence, enabling it to detect subtle linguistic patterns and deceptive cues used in phishing attempts. In recent studies, BERT has been applied to phishing detection by transforming message content into 768-dimensional contextual embeddings using the [CLS] token, which captures the overall semantics of the text. These embeddings, when combined with other extracted features such as URLs, email addresses, or phone numbers, are used as inputs for neural network classifiers to distinguish phishing messages from legitimate ones. Compared to traditional text classification techniques, BERT has demonstrated superior performance in identifying context-based phishing strategies, making it a valuable AI tool for building intelligent systems that can proactively detect and prevent social engineering-based cyber threats.



**Figure 7: Transformer Model Architecture (Jain et al., 2025)**

### Reinforcement Learning Approaches

Reinforcement learning (RL) is one of the sub-domains of machine learning. The goal is to let the agent learn how to act based on the environment state to maximize the expected long-term rewards, where the learning problem can usually be modeled as Markov decision problems (MDPs). Figure 8 shows the interactive feedback loop between the agent and the environment. (Wang et al.,2020)



**Figure 8: Interaction between the agent and the environment: at each time step, after the agent observes the environment, it chooses an action according to its policy. After the action is executed, the environment gives a reward signal to the agent and transit to a new state.**

Reinforcement learning (RL) agents are generally trained in episodes, each consisting of a certain number of steps. Given an episode, the sequence of states, actions, and rewards builds the trajectory or rollout of π. Let *k* be the index assigned to an episode; the *cumulative discounted reward* is defined as = . Then, the objective function to be optimized can be indicated as and the maximization problem, which the agent tries to solve, aims at finding for all and .

#### Q- Learning

Q-learning is a value-based reinforcement learning algorithm. The goal of Q-learning is to learn the optimal action-selection policy for an agent interacting with an environment. The agent learns this by updating a table of values called the Q-table where each entry represents the value of taking a particular action in each state (Kovalchuk, 2024).

The Q-learning algorithm uses the following formula to update the Q-value for a state-action pair based on figure below:

A math equation with black text

AI-generated content may be incorrect.

**Figure 9: Q – Learning equation (Kovalchuk, 2024)**

Where:  
- Q(s, a) is the Q-value for state s and action a.  
- alpha is the learning rate, controlling how much new information overrides old information.  
- R is the immediate reward for taking action a in state s.  
- gammais the discount factor, representing the importance of future rewards.  
- max\_a Q(s’, a’) is the maximum Q-value for the next state s’, representing the best possible reward achievable from that state.

#### Deep Q-Learning

Deep Q-Learning or Deep Q Network (DQN) is an extension of the basic Q-Learning algorithm, which uses deep neural networks to approximate the Q-values. Traditional Q-Learning works well for environments with a small and finite number of states, but it struggles with large or continuous state spaces due to the size of the Q-table. Deep Q-Learning overcomes this limitation by replacing the Q-table with a neural network that can approximate the Q-values for every state-action pair (Amin, 2024).

A diagram of a network

AI-generated content may be incorrect.

Figure 10: Structure of DQN

**Key Concepts of Deep Q-Learning**

1. **Q-Function Approximation**: Instead of using a table to store Q-values for each state-action pair, DQN uses a neural network to approximate the Q-values. The input to the network is the state, and the output is a set of Q-values for all possible actions.
2. **Experience Replay**: To stabilize the training, DQN uses a memory buffer (replay buffer) to store experiences (state, action, reward, next state). The network is trained on random mini-batches of experiences from this buffer, breaking the correlation between consecutive experiences and improving sample efficiency.
3. **Target Network**: DQN introduces a second neural network, called the target network, which is used to calculate the target Q-values. This target network is updated less frequently than the main network to prevent rapid oscillations in learning.
4. **Bellman Equation in DQN**: The update rule for DQN is based on the Bellman equation, like Q-Learning:



Figure 11: Bellman equation i

Where:

* θ are the weights of the main Q-network,
* θ− are the weights of the target Q-network,
* s is the current state,
* a is the action taken,
* ris the reward received,
* s′ is the next state,
* maxa′Q​Q(s′,a′) is the maximum Q-value for the next state.

## Critical Review

Phishing attacks are a prevalent form of social engineering that exploit human vulnerabilities to gain unauthorized access to sensitive information. Attackers impersonate legitimate entities to deceive users into disclosing credentials, financial information, or other confidential data. Traditional rule-based detection methods have proven inadequate against the evolving sophistication of phishing techniques. Consequently, Artificial Intelligence (AI) approaches, including Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL), have been increasingly explored to enhance phishing detection capabilities. This critical review examines and compares various AI methods based on recent research, focusing on features used, mathematical foundations, comparative performance, and justifications grounded in empirical findings.

### Features and Mathematical Formulations

#### Machine Learning

Traditional ML models rely on carefully engineered features extracted from emails, URLs, or websites. Common features include URL length, number of dots, presence of '@' symbol, HTTPS usage, domain age, and number of redirects (Rawshon Ferdaws & Majd, 2024).

1. **Support Vector Machine (SVM)**

SVMs find the optimal hyperplane that separates phishing from legitimate data. The optimization goal is:

subject to

where *w* is the weight vector, are the input vectors, are the labels, and *b* is the bias.

1. **Random Forest (RF)**

Random Forest builds multiple decision trees and aggregates their predictions. Each tree splits data based on criteria such as Gini Impurity:

where is the probability of class *i* at a node.

Traditional machine learning techniques such as Random Forest (RF) and Support Vector Machine (SVM), rely heavily on manually extracted features from URLs, email headers, or web content. These features are typically selected based on domain knowledge and include indicators such as URL length, number of subdomains, presence of IP address in URL, use of HTTPS protocol, frequency of suspicious keywords, and domain registration age. According to the paper *“Phishing URL Detection Using Machine Learning and Deep Learning” (2023)*, these features are effective because many phishing URLs tend to share common structural patterns.

The main advantage of traditional ML models lies in their **interpretability and simplicity**. Techniques like Random Forest not only provide high accuracy but also allow feature importance ranking, helping researchers understand which indicators contribute most to the prediction. Moreover, models like SVM are relatively easy to train and can be effective with smaller, well pre-processed datasets. This makes them suitable in environments with limited computational resources or when explainability is crucial as emphasized in *“Analysis of ML Techniques for Phishing Detection” (2022*).

However, these methods face significant **limitations when handling complex or unstructured data**. Manual feature engineering is time-consuming and may fail to capture nuanced patterns in phishing content, particularly when attackers modify phishing strategies slightly to evade detection. Furthermore, these models may not generalize well across datasets or domains, as highlighted in *“Comparison of Machine Learning Algorithms for URL Phishing Detection”, 2022*. They are also less robust against sophisticated obfuscation techniques used in modern phishing campaigns.

#### Deep Learning

Deep learning models automatically learn high-level features from raw data such as text, HTML source code, and URLs (Application of Natural Language Processing for Phishing Detection, 2022).

1. *LSTM*

Long Short-Term Memory (LSTM) networks, a type of Recurrent Neural Network (RNN), have proven highly effective in phishing detection tasks due to their ability to model sequential dependencies and temporal patterns in data. LSTMs are especially suitable for processing natural language text, making them valuable for detecting phishing emails or malicious messages that rely on deceptive language structures. These models overcome the vanishing gradient problem found in traditional RNNs through their internal gating mechanisms: the **input gate**, **forget gate**, and **output gate**, which control the flow of information and preserve context over long sequences.

Mathematically, an LSTM unit at time step ttt involves the following core equations:

A black background with white text

AI-generated content may be incorrect.

These mechanisms allow the LSTM to **retain or forget information selectively**, making it particularly useful in detecting phishing messages where critical indicators (e.g., urgency cues, mismatched domain names, suspicious requests) may appear at various positions in the message.

In the study *“Application of Natural Language Processing for Phishing Detection” (2022)*, an LSTM-based model was trained on phishing email datasets using pre-trained word embeddings (e.g., GloVe). The model successfully captured contextual nuances that distinguish phishing from legitimate communication. The LSTM demonstrated higher precision and recall compared to baseline models like logistic regression and SVM.

One significant advantage of LSTM is its **capability to capture sequential and syntactic dependencies** without manual feature engineering. This makes it particularly effective in phishing detection when the attack relies on language patterns rather than structural anomalies in URLs or email headers. The use of word embeddings further enhances the model’s ability to understand semantic relationships between words.

However, LSTMs also come with **certain limitations**. They are computationally expensive to train, especially on large-scale datasets. Their architecture requires a careful tuning of hyperparameters such as the number of layers, hidden units, and learning rate. Moreover, like other deep learning models, LSTM networks lack transparency—making them harder to interpret or explain in terms of individual predictions. This "black-box" nature could limit adoption in high-stakes environments where auditability is essential.

Despite these challenges, the literature shows that LSTM-based phishing detectors provide a strong balance between performance and robustness. They can generalize better to previously unseen patterns, particularly in email or message-based phishing attacks, making them a powerful tool in the fight against social engineering threats.

1. *BERT*

BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. (2018), has revolutionized Natural Language Processing (NLP) by introducing deep bidirectional context-aware language modeling. In phishing detection, BERT is leveraged to understand the semantic structure and contextual relationships in email or message content, providing robust performance in classifying malicious intent based on text.

Unlike LSTM, which reads sequences from left to right (or bidirectionally in advanced versions), **BERT is inherently bidirectional**, meaning it considers the entire sentence context (both left and right) at every layer. This capability is particularly useful for detecting subtle linguistic indicators of deception in phishing emails—such as urgency, impersonation, or requests for sensitive information.

*Mathematical Model of BERT*

BERT is built upon the Transformer architecture, which relies heavily on self-attention mechanisms. The core component of BERT is the multi-head self-attention operation defined as:

Attention (Q, K, V) = softmax

Where:

* Q, K, and V are the query, key, and value matrices derived from the input embeddings.
* is the dimensionality of the key vectors (used for scaling).
* The output is a weighted sum of the value vectors, where weights are determined by the similarity between the query and key vectors.

Multiple attention heads capture different relationships simultaneously across the text, helping BERT understand phishing intent embedded in diverse sentence structures.

1. *Advantages*

BERT is exceptionally accurate in phishing detection tasks due to its semantic-level reasoning. Studies from your dataset (e.g., “Deep Learning-Based Phishing Detection Using BERT”) show that BERT outperforms CNNs, LSTMs, and SVMs across all key metrics. It adapts easily to new tasks through fine-tuning and is robust across different phishing types (spear-phishing, clone-phishing, etc.).

1. *Limitations*

The biggest drawback of BERT is its resource cost. Training or even fine-tuning BERT requires significant GPU memory and processing time. Moreover, like LSTM, BERT is not interpretable—its decisions are not easily mapped to human-readable rules. This opacity can be a challenge for compliance and audit trails in cybersecurity.

#### Reinforcement Learning

Reinforcement Learning (RL) has been explored recently for adaptive phishing detection. In RL, agents learn optimal detection strategies through interactions with the environment, receiving rewards for correct classifications.

1. Deep Q-Network (DQN)

DQN uses deep neural networks to approximate the Q-function:

where:

* *s* is the current state (e.g., features of a URL or email),
* *a* is the action (predict phishing or not),
* *r* is the reward,
* γis the discount factor.

By maximizing cumulative rewards, DQNs can adaptively improve detection strategies against evolving phishing tactics (Detecting Phishing Websites through Deep Reinforcement Learning, 2022).

Reinforcement Learning (RL) models represent a newer and more dynamic approach to phishing detection. Unlike ML and DL, which rely on fixed datasets and offline training, RL systems interact with an environment and learn optimal strategies through trial and error. In the context of phishing detection, an RL agent might explore different classification policies for URLs or emails, receiving rewards for correct classifications and penalties for errors. The paper *“Detecting Phishing Websites Through Deep Reinforcement Learning” (2022)* proposed a Deep Q-Network (DQN)-based model that learned to distinguish phishing websites by simulating a browsing agent that collected observations of website features.

The **main advantage of RL** lies in its **adaptive capability**. Because phishing tactics evolve over time, static models may become obsolete, while RL agents can continuously learn from feedback and adjust their detection strategies. This makes RL particularly attractive in adversarial environments where attackers change behavior to avoid detection. The paper *“Adaptive Phishing Detection Using RL” (2022)* emphasized that RL outperformed supervised methods in long-term deployment, as it could adapt to new phishing styles do not present in the original training data.

Nonetheless, RL is still **experimental and less mature in this domain**. One of the primary limitations is the high training time and the need for a simulated or real-time environment where the agent can interact and receive meaningful rewards. Designing a reward function that accurately reflects phishing detection success is complex, and poor design can lead to suboptimal learning. Moreover, RL models require more careful tuning of hyperparameters like learning rate, discount factor, and exploration strategies. According to *“Challenges in Applying Reinforcement Learning for Phishing Defense” (2023)*, stability and reproducibility remain open issues in practical applications.

### Comparative Summary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Technique** | **Feature Type** | **Model Complexity** | **Interpretability** | **Accuracy** | **Real-Time Usability** | **Typical Use Case** |
| Traditional ML | Manual | Low | High | Moderate | High | URL/email phishing using structured features |
| LSTM | Sequence-based | Medium–High | Low | High | Medium | Text-based phishing detection |
| BERT | Contextual language | Very High | Low | Very High | Low | Deep semantic email/message classification |
| Reinforcement Learning | Environment-driven (policy/state) | High | Moderate–Low | Variable (task-dependent) | Low–Medium | Adaptive phishing defense, behavior learning |

### Justification

Numerous empirical studies justify the growing dominance of deep learning (DL) models in phishing detection over traditional machine learning (ML) and reinforcement learning (RL), especially in contexts involving complex, text-based social engineering attacks.

Deep learning architectures, particularly **Convolutional Neural Networks (CNNs)** and **Long Short-Term Memory (LSTM)** networks, have proven effective in capturing both **semantic** (meaning) and **structural** (positional/pattern-based) features in phishing content. The study *“Application of NLP for Phishing Detection” (2022)* highlights how CNNs can extract relevant word patterns in messages, while LSTMs capture long-range dependencies — such as repeated urgency tones or inconsistencies across sentences. These models do not require handcrafted features, making them highly adaptable to evolving phishing content and superior in detecting obfuscated, grammar-correct phishing messages that traditional models may miss.

On the other hand, **traditional ML models** like **Random Forests** remain relevant when a rich set of engineered features is available. The paper *“Analysis of Machine Learning Techniques for Phishing Detection” (2022)* demonstrates that when lexical, URL-based, and domain-based features are extracted, tree-based models achieve competitive results with significantly lower training time and greater interpretability. This makes them attractive in systems where transparency and fast deployment are priorities. However, these models are heavily dependent on the quality and availability of the feature set, which becomes a limitation when facing zero-day or linguistically sophisticated attacks.

**Reinforcement Learning (RL)** introduces a different paradigm. Rather than being trained on static labeled data, RL agents learn to optimize behavior through rewards based on environmental interaction. In phishing detection, this means adapting policies to recognize evolving attacker behavior. The study *“Detecting Phishing Websites through Deep Reinforcement Learning” (2022)* illustrates the use of Deep Q-Networks to train agents that improve their detection strategies over time. While RL models are still emerging in this space and show slightly lower accuracy compared to DL counterparts, their **ability to continuously adapt** makes them promise for future **dynamic defense systems**, especially in live email systems or phishing-aware chatbots.

An emerging trend is the development of **hybrid models** that integrate both ML and DL components. The study *“Phishing Detection by Integrating Machine Learning and Deep Learning” (2023)* reports that such models benefit from the **interpretability and structure-awareness of ML**, while also leveraging the **contextual and semantic depth of DL models**. For example, combining URL-based ML classifiers with deep textual analysis using LSTM or BERT improves overall robustness, especially in detecting multi-layered phishing campaigns that span both surface-level and deep content features.

In summary, research supports the use of deep learning models particularly LSTM and BERT as the current state-of-the-art for phishing detection, especially for text-heavy, content-driven social engineering attacks. Traditional ML models still offer practical benefits when structured features are available, and reinforcement learning holds future promise for adaptive and evolving defense systems. The trend toward hybridization suggests that no single technique is universally optimal, but deep learning remains the most effective core strategy in contemporary academic and practical implementations.

## Project Solution

### Overview of Proposed Solution

The proposed project solution aims to detect phishing-based social engineering attacks through a deep learning-driven approach that leverages the linguistic structure and contextual patterns of phishing messages. Based on the critical review and justifications, the solution will focus on applying a fine-tuned BERT (Bidirectional Encoder Representations from Transformers) model to identify phishing content in email or message text. This model will be supported by a comparative baseline using LSTM and a traditional ML classifier (e.g., Random Forest) to measure effectiveness and performance.

### System Architecture

The architecture of the proposed solution consists of the following key components:

1. **Data Collection**

* Dataset includes phishing and legitimate emails or messages from publicly available sources (e.g., PhishTank, Kaggle, or datasets referenced in research papers).
* Data will be pre-labeled to train supervised models.

1. **Data Preprocessing**

* Tokenization, lowercasing, removal of stop words (for LSTM & ML models).
* For BERT: Apply BERT-specific tokenizer (bert-base-uncased) that preserves subword tokens and casing appropriately.
* Class balancing using techniques like SMOTE or undersampling (if required).

1. **Feature Engineering**

* **For ML (e.g., Random Forest)**: Extract lexical features (e.g., presence of “urgent,” domain patterns, punctuation usage, number of links).
* **For LSTM**: Use embedded word sequences.
* **For BERT**: Direct use of pre-trained embeddings via transformer layers (no manual feature engineering).

1. **Model Development**

* **Baseline ML Model**: Train Random Forest on structured features.
* **LSTM Model**: Train an LSTM on preprocessed textual sequences.
* **BERT Model**: Fine-tune bert-base-uncased on the phishing dataset.

1. **Evaluation Metrics**

* Accuracy, Precision, Recall, F1-Score, and AUC-ROC.
* Confusion matrix for each model to visualize performance on both phishing and legitimate classes.
* Compare performance across all three models to validate the superiority of BERT.

1. **Deployment (Optional for FYP scope)**

* Integrate the best-performing model (likely BERT) into a simple web-based UI for demonstration purposes (e.g., Flask or Streamlit).
* Users can input email text and receive a phishing probability score and decision.

### Justification for Model Choice

The selection of BERT is based on strong research support for its superior performance in phishing detection due to its ability to understand contextual cues and language semantics. As phishing emails often mimic legitimate messages using subtle textual patterns, traditional ML models fail to generalize across new tactics. LSTM provides improved performance over ML by capturing sequence patterns, but BERT outperforms both by analyzing full context and syntax bidirectionally. Including Random Forest and LSTM models provides a comparative benchmark, allowing the study to justify BERT’s performance with empirical results.

### Anticipated Benefits

* High **detection accuracy** through deep contextual understanding.
* Reduced **false positives**, especially in emails with legitimate-sounding text.
* Scalability **and adaptability** through fine-tuning with new datasets.
* Educational **impact** by enabling an in-depth comparative study across AI techniques in cybersecurity.

### Limitations and Future Work

* Computational **cost** for BERT training/fine-tuning, which may require access to GPU resources (e.g., Google Colab).
* Limited **real-time testing** unless deployed in a simulated environment.
* Future work may explore **reinforcement learning** models for adaptive phishing defense or integrate **multimodal data** (e.g., combining email content with sender metadata or URL analysis).

## Summary

This chapter presented a detailed review of Artificial Intelligence (AI) approaches used to detect phishing-based social engineering attacks. The analysis covered three main categories: Traditional Machine Learning (ML), Deep Learning (DL), and Reinforcement Learning (RL). Each method was examined based on its core features, theoretical foundation, performance outcomes, and relevance to phishing detection.

Traditional ML models such as Random Forest and Support Vector Machines are widely used due to their simplicity and interpretability. These models perform well when trained on well-engineered lexical, structural, and URL-based features. However, their effectiveness diminishes when facing sophisticated phishing attempts that rely heavily on natural language and mimicry techniques.

Deep Learning models, particularly Long Short-Term Memory (LSTM) networks and BERT, have shown superior performance in recent research. LSTM models effectively capture sequential dependencies in text, making them suitable for analyzing the flow of phishing messages. BERT, with its bidirectional transformer architecture, offers a deeper contextual understanding of message semantics and achieves state-of-the-art performance in phishing email classification tasks.

Reinforcement Learning models are relatively new in this field but present a promising direction. Their ability to learn from dynamic environments and adjust to evolving phishing strategies could be valuable in real-time systems, although practical implementation challenges remain.

Justifications from empirical studies highlight the advantages of deep learning—especially BERT—in detecting phishing attacks, while traditional ML models still serve as useful benchmarks. The potential of hybrid approaches was also noted, where integrating ML and DL methods can lead to more robust systems. These findings form the foundation for the project’s proposed solution, which will focus on deep learning (particularly BERT and LSTM), supported by comparative evaluation against traditional ML methods.

# METHODOLOGY

## Introduction

## Methodology

### Requirement Analysis

#### Hardware Requirement

A laptop is used as a workstation for all tasks, from researching to documenting. The laptop specifications are shown in the table below.

Table 3.2‑1: Hardware Requirement

|  |  |
| --- | --- |
| **Specification** | **Description** |
| Processor Type | AMD Ryzen 5 5600U with Radeon Graphics 2.30 GHz |
| Operating System | Windows 11 Version 21H2 |
| Operating System Type | 64-bit operating system, x64-based processor |
| RAM | 8.00 GB (7.35 GB usable) |
| Storage | 476 GB |
| Display Resolution | 1920 x 1200 |

#### Software Requirement

This project's development includes the usage of some software. The software used in this project is listed in the table below.

Table 3.2‑2: Software Requirement

|  |  |
| --- | --- |
| **Software** | **Description** |
| Windows 10 | An environment of operating system used for project execution. |
| GNS3 | Software used to execute the simulation/experiment based on the topology selected. |
| VMware Workstation Pro | Software to run virtual machines |
| Wireshark | Capture network traffic for analysing purpose. |
| Microsoft Word 365 | Software used to complete the project reporting and documentation. |
| Microsoft Excel 365 | Software used to sort the data according to attributes and instances, also to create graph. |
| Microsoft PowerPoint 365 | Software used to create charts or framework. |

### System Design

### Implementation

### Testing

### Deployment

### Maintenance

During this phase, if the testing fails, some maintenance is made to determine the cause and resolve the problem. To locate the mistake, every configuration made to measure the metrics are examined. The testing stage then is performed several times until a better result is attained. As a result, the maintenance phase occurs during testing and can also be applied if the project encounters an error. Recommendations to improve IoT performance may be made based on the simulation results. This section also discusses the difficulties encountered during the implementation process, from setting up the environment to gathering findings.

## Project Schedule and Milestones

### Project Milestones

A milestone is a project marker that denotes a shift or stage of progress. Thus, project milestones is essential in keeping track of upcoming events or goals across the timeline.

|  |  |  |
| --- | --- | --- |
| **Week** | **Phase** | **Activity** |
| 1-6 | Requirement Analysis | * Gather information regarding IoT protocol. * Studies on related work and previous research of performance analysis of IoT protocol. * Analyse the methods used by previous researchers to do their study. * Studies on hardware and software used to run the experiment. |
| 7-14 | System Design | * Information collection and analysis. * Project design – where to implement and topology selection. * Project demonstration and report submission to supervisor and evaluator. |
| ***SEM BREAK*** | | |
| 15-18 | Implementation | * Installation of GNS3 and ISO used. * Setup selected topology in simulator, GNS3. * Configure nodes, switches, servers based on the needs. * Configure topology so it can measure the chosen metrics (bandwidth utilisation, latency, and throughput) * Integrate VMware with topology in GNS3 |
| 19-22 | Testing | * Test the configuration done. * Monitor performance for each protocol at different topology. |
| 23-25 | Deployment | * Describe the results gain from the simulation. * Critical review of findings from the simulation done. |
| 26-28 | Maintenance | * Provide recommendation to improve IoT performance. * Project demonstration and final report submission to supervisor and evaluator. |

### Project Gantt Chart

Gantt Charts provide a thorough overview of the project from start to finish, as well as all the activities required to complete the project. It aids in demonstrating how far the tasks have progressed.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PHASE** | **WEEK** | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| Requirement Analysis |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| System Design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Maintenance |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Summary

# ANALYSIS AND DESIGN

## Introduction

This chapter briefly outline the project's design that is going to be implemented in order to make the project clearer. This chapter provides logical and physical design with explanation. Possible scenarios in this project are also stated. In addition, the simulation's topology design is discussed during this phase. As a result, this chapter provides a good understanding of the specific topology that is implemented in the next chapter.

## Summary

To summarize, this chapter is vital as it contains the clear idea on how to develop the project, in this case, to analyse the performance of IoT protocol. The chapter goes through system architecture, physical and logical design of the simulation and also includes the proper measurement of metrics. The upcoming phase, implementation phase, must be carried out using the analysis and design achieved in this chapter.

# IMPLEMENTATION

## Introduction

## Summary

# discussion

## Introduction

## Result and Analysis

## Summary

# CONCLUSION

## Introduction

## Project Summarization

## Project Contribution

## Project Limitation

## Future Works

## Summary

# references