National College of Ireland

Enhancing Phishing Detection Using O3-Mini Chain-of-Thought Reasoning with GPT-4o

MSc Research Practicum
MSc in Artificial Intelligence

Bharath Reddy Choudary
Student ID: X23353414

School of Computing
National College of Ireland

Supervisor: Mohit Garg

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Bharath Reddy Choudary		
Student ID:	X23353414		
Programme:	MSc in Artificial Intelligence	**Year:**	2024-2025
Module:	MSc in Research Practicum		
Supervisor:	Mohit Garg		
Submission Due Date:	15/09/2025		
Project Title:	Enhancing Phishing Detection Using O3-Mini Chain-of-Thought Reasoning with GPT-4o		
Word Count:	9328	**Page Count:**	27

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Bharath Reddy Choudary

Date: 14/09/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhancing Phishing Detection Using O3-Mini Chain of-Thought Reasoning with GPT-4o

Bharath Reddy Choudary
X23353414

Abstract

The phishing attacks are still highly sophisticated using advanced social engineering techniques that outplay traditional detection methods. While deep learning approaches shown some hope, they act more as black boxes, lacking interpretability required for security applications. This paper discusses a new way of phishing detection enhancement through enriched GPT-4o and chain-of-thought reasoning by OpenAI's o3-mini model.

The experimental design is a three-stage one based on 10,000 balanced emails from the corpus of Enron. In the first phase, o3-mini generates structured argumentation chains assessing five aspects of security, i.e., sender legitimacy, linguistic characteristics, employment of methods of social engineering, technical characteristics, and risk assessment. The argumentation chains are also used for fine-tuning of GPT-4o, thus having a model that makes classification judgments simultaneously and also generates well-explained descriptions. The testing pits four setups against each other: baseline GPT-4o, few-shot GPT-4o, reasoning-augmented fine-tuned GPT-4o, and o3-mini in isolation.

The result indicates that the method enhanced by rationality achieves 99.0% accuracy, an increment by 18.4 percentage points from baseline GPT-4o (80.6%), while exhibiting a 94.5% overall error minimization. Precision and recall are balanced equally at 99.0% by the fine-tuned model, and no classification bias is observed in comparison to baseline configurations.

The proposed method achieves 99.0% accuracy compared to the baseline's 80.6%, reducing errors by 94.5% and maintaining a consistent score of 99.0% for both recall and precision.

1 Introduction

Phishing attacks are the most long-standing cyber security challenges for the individuals and organizations across the globe. Along with the advanced detection technology, there are more sophisticated and credible means of social engineering and exploiting human behavior being used by phishing attacks. Traditional detection technologies, although successful with well-known patterns, struggle to address new methods of attacks continuously being developed. This project investigates a new direction for enhancing phishing detection capability with an addition of GPT-4o with chain-of-thought reasoning from OpenAI's o3-mini model.

The phishing threat evolved from simplistic fraudulent emails to sophisticated multi-layered attacks combining technology-driven exploitation with psychological manipulation. Recent systematic reviews identify that whilst methods using machine learning are promising, deep learning approaches offer superior performance with 3-7% enhancements in conventional approaches (Kyaw et al., 2024). However, an overriding weakness of current methods identified is a deficiency in interpretability of classification results, particularly for applications in security where understanding of reasoning for detection of threats is of utmost significance (Salloum et al., 2021).

The arrival of Large Language Models (LLMs) revealed new possibilities for natural language understanding for security uses. Multimodal model by OpenAI, GPT-4o, has advanced pattern recognition but gives in to new methods of deception (OpenAI, 2024). Concurrently, improved reasoning models like o3-mini through chain-of-thought approaches for decomposing complex problems into sequential procedures open a potential solution for that gap.

The motivation for this work lies in three main gaps of existing phishing detection literature. First, although there is a ocean of work examining deep learning approaches for phishing detection, comparatively less have looked at, in a systematic manner, explicit chains of reasoning for increasing model efficiency alongside interpretability. Secondly, a transition from black-box neural networks to explicit chains of reasoning fulfills a basic need in security applications in that decisions must be explainable and auditable. Thirdly, although development of phishing tactics is proceeding at a very rapid rate, detection systems must generalize beyond patterns learned from the training data through logical rather than pattern-matching reasoning.
The consumers of this piece of work vary from research use to practice use. The security researchers gain knowledge about detection approaches augmented by reasoning, practitioners gain a more explainable and dependable method of protecting email and those that focus on sensitive content in particular gain the benefits of the extra interpretability of threat detection decision-making.

OpenAI's o3-mini model from January 2025 is a chain-of-thought focused reasoning model for decomposing hard analysis problems into properly structured reasoning steps. It is a milestone for explainable AI for explicit decision-making and explicit chains of reasoning. OpenAI's GPT-4o mini from July 2024 is OpenAI's computationally light 128K window context small language model selected for its best trade-off among computation efficiency and performance. They support advanced reasoning generation and fiscally responsible fine-tuning within realistic computation constraints.
The primary research question behind this investigation is: **How does the integration of o3-mini-generated chain-of-thought reasoning enhance GPT-4o's capability to detect phishing emails?**

The objectives of this study are outlined as follows:

1. To investigate the extent to which structured chains of reasoning, when integrated with GPT-4o, enhance phishing detection performance beyond conventional control-based methods.
2. To compare the effectiveness of reasoning-augmented models with baseline implementations under zero-shot and few-shot learning configurations, thereby assessing their adaptability and generalization capabilities.
3. To evaluate the interpretability improvements afforded by explicit reasoning chains in classification decisions, with a focus on their contribution to transparency and trustworthiness in phishing detection systems.
4. To analyze the trade-offs in computation required for achieving higher detection accuracy by means of reasoning augmentation and consider the implications for processing scalability and efficiency in real-world applications.

The evaluation for this work is based upon three core criteria: achieving statistically significant improvements of phishing detection performance, fostering structured chain reasoning for enhanced interpretability, and preserving computationally efficient real-world deploy ability. To attain these goals, the new approach weaves o3-mini chain reasoning together for the examination of five critical phishing indicators per email. This approach enriches the dataset by embedding structured reasoning, leading to more accurate and interpretable classification outcomes.

The report is organized to guide readers through the development and evaluation of reasoning-augmented phishing detection using GPT-4o, beginning with a comprehensive literature review in Chapter 2, followed by detailed methodology and data preparation in Chapter 3. Chapter 4 outlines the system design, while Chapter 5 details the implementation process. Experimental results are presented in Chapter 6, with findings and limitations discussed in Chapter 7. Finally, Chapter 8 concludes the study and proposes directions for future research.

The introduction establishes the basis for exploring how structured reasoning can enhance the accuracy and interpretability of phishing detection systems. This new approach is explored in a systematic way in later chapters with extensive experimentation as well as analysis.

2 Related Work

The literature responding to phishing detection with computational methods significantly expanded during recent years with particular emphasis on deep learning and machine learning approaches. In this survey, four thematic groups of literature are critically examined: systematic reviews of phishing detection methods, traditional machine learning methods, extensions of deep learning methods, and the emerging new direction of chain-of-thought reasoning for language models. Each of the thematic groups presents significant insights with clear weaknesses that motivate the ongoing research.

2.1 Systematic Reviews of Phishing Detection Approaches

A few researchers have conducted in-depth reviews for consolidating a summary of the existing state of phishing detection research. A PRISMA-guided systematic literature review by Salloum et al. (2022) explored phishing detection from emails using deep learning, considering 100 papers

from an original 1,125 publications after thorough quality appraisal. According to the review, while a vast amount of research examined making use of machine learning, very few explored deep learning techniques systematically for detecting phishing emails (Salloum et al., 2022). This neglect stands out particularly given the proficiency with which deep learning techniques have demonstrated to natural language processing tasks.

The systematic review brought into view prominent trends in methods, with support vector machines appearing in 50 papers, Naïve Bayes in 33 papers, and decision trees in 29 papers for traditional methods (Salloum et al., 2022). For deep learning methods, Recurrent Neural Networks (RNN), Standard Neural Networks (NN), and Convolutional Neural Networks (CNN) were applied with differential frequencies (Salloum et al., 2022). Another significant finding was that 83 experiments utilized accuracy as their principal measure, followed by F1-measure in 38 experiments, with about 35 experiments for each of confusion matrix, recall, and precision (Salloum et al., 2022).

Similar findings were echoed in another systematic review by Kyaw, Gutierrez and Ghobakhlou (2024), which followed PRISMA guidelines and analyzed 33 selected papers specifically focusing on deep learning techniques for phishing email detection. This review also noted that traditional methods may face difficulties in generalizing to unseen and complex patterns in data, though they are computationally inexpensive and simple to deploy (Kyaw, Gutierrez & Ghobakhlou, 2024).

The benefits of these reviews include thorough coverage and strict adherence to traditional systematic review protocols. The quality assessment step, which uses a 7-point checklist with scoring criteria, makes sure that only studies of high quality are looked at (Salloum et al., 2022). The main problems with reviews, though, are that they focus more on quantity than on depth of analysis. There are nice collections of methods, but not a lot of critical thought about why some methods can't be used on new attack vectors.Also, there is a disturbing trend of research towards favoring accuracy measures without appropriate class imbalance treatment for the phishing detection cases (Salloum et al., 2022; Kyaw, Gutierrez & Ghobakhlou, 2024).

2.2 Traditional Machine Learning Approaches

Since the conventional machine learning methods have also been fruitfully applied for phishing detection, they have served notable baselines for evaluating the performance of the subsequently proposed methods. Using Nazario phishing corpus (42 studies), Spam-Assassin Public Corpus (40 studies), and Enron dataset (23 studies), the feature-based methods have been established as tractable. Text feature extractors like bag-of-words (BoW), information gain (IG), and Word2vec have gained favour, and BoW and IG have appeared in 10 and nine studies, while Word2vec has also appeared in nine studies. Salloum et al. (2022) conducted a survey of detection methods of phishing using NLP and found that TF-IDF was employed as the most effective preprocessing method, in favour of word embeddings, and SVM was employed as the most effective algorithm. The exploration by Al-Subaiey et al. (2024) resulted in an impressive 99% F1-score using SVM and TF-IDF preprocessing, and this illustrates well-tuned traditional methods can be very efficient for selective datasets.

Classical machine learning (ML) methods are very good at having very low computation overhead and being easy to understand. Feature importance analysis and information gain are techniques that enable data analysts to discern the attributes that most efficiently distinguish

phishing emails and provide clarity in decision-making. These strengths come with huge costs. Because their features are hand-made, classical ML models can't learn the subtle language features that are often used in advanced phishing attacks. Also, the fact that some features can't be changed makes the models less flexible and means that they need to be retuned very often to stay useful when attack methods change. Inconsistencies in the results of best feature selection experiments show that these methods don't work well on all datasets.

2.3 Deep Learning Advancements in Phishing Detection

Moving from traditional machine learning to deep learning is a big step forward in the fight against phishing. According to Kyaw et al. (2024), deep learning-based models like Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) show a 3–7% improvement over traditional methods, especially when used with complex datasets. This breakthrough is solely due to the intrinsic capability of deep learning to autonomously derive high-level feature representations from the raw data inherent in email content, eliminating the need for manually crafted representations.

This observation indicates that RNNs are proficient in learning sequential patterns within email content, making them particularly effective for understanding context and flow. CNNs are great at finding a localised structure and are good at picking up on the small hints that are often hidden in phishing emails. The Adam optimiser, which was used 26% of the time, was the most common optimisation method. The sequential minimal optimisation (SMO) method, which was used 21% of the time, was mostly used to figure out how important a word was in the text datasets. These models are better than other ones that use very dynamic linguistic datasets and advanced social engineering techniques.

Deep learning methods have the advantages of automatically learning high-level features without having to handcraft them and the potential to learn non-linear data relationships. Hierarchical learning finds subtle patterns that rule-based or traditional machine learning methods would miss. However, there are still some major problems. Salloum et al. (2021) specifically pointed out that model interpretability is a problem, even though it works better. This is a necessary condition for security applications where it is important to understand why something is classified. Because deep neural networks are "black boxes," it's hard to understand why they classify some emails as phishing, which makes people less sure about automated solutions. Also, deep learning models require much larger datasets for computation and learning in comparison to traditional approaches.

2.4 Chain-of-Thought Reasoning and Language Models

The arrival at chain-of-thought (CoT) prompting is a qualitative shift for how high-level problems in reasoning are tackled by language models. Chain-of-thought prompts made their first appearance in Wei et al. (2022), when it was demonstrated that large language models would gain 20-40% performance boosts for a suite of reasoning benchmarks when presented intermediate reasoning steps rather than base-line prompting.

Chain-of-thought (CoT) prompting helps break down complex problems into logical steps. Research shows it improves performance in arithmetic tasks and supports generalization in common sense reasoning. In symbolic reasoning, CoT enables models to handle longer sequences and out-of-distribution inputs, suggesting deeper reasoning beyond pattern matching.

The key advantage of chain-of-thought (CoT) approaches is their ability to make model reasoning explicit and auditable. By outlining step-by-step logic, CoT reduces the interpretability gap found in both traditional deep learning and language model methods. This transparency allows for the identification of logical flaws and biases in decision-making. However, despite these strengths, current implementations still face certain limitations. While chain-of-thought prompting emulates human reasoning processes, it isn't certain whether neural networks do indeed "reason" as opposed to spinning plausible reasoning tales. Being able to create false reasoning pathways leading to both right as well as wrong answers creates challenges for reliability. Being computationally too costly for high-throughput applications such as email filtering is another limitation of generating sophisticated reasoning chains.

2.5 Data Augmentation Techniques in NLP

Data augmentation methods were found to be significant techniques for improving model robustness for natural language tasks, with direct application for phishing detection. Research demonstrates that a combination of multiple augmentation methods can be a superior performing technique. Some research indicates a benefit of combining paraphrasing-based techniques, for instance, combining thesaurus methods with semantic embeddings, or combining semantic embeddings with masked language models (MLMs).

Simple and task-independent unsupervised data augmentation techniques have gained widespread popularity. Such software packages as EDA (Easy Data Augmentation), which is an aggregation of synonym substitution, random insertion, random swap, and random deletion, have been particularly successful. Furthermore, the noising and back-translation based unsupervised methods like UDA have also been fully applied for most classification tasks.

Data augmentation plays a vital role in improving model generalization and handling data limitations, especially in phishing detection. By generating new instances from existing samples, it helps models learn robust features that are less sensitive to superficial textual changes. However, its effectiveness in security contexts is limited. Augmented data often fails to capture the complexity and subtlety of real-world phishing tactics, leading to models that perform well on synthetic inputs but struggle with actual threats. Maintaining semantic coherence amid large textual variations is particularly challenging, as even minor word changes can shift the intent in phishing emails, making reliable detection more difficult. Optimization of augmentation techniques including tuning hyperparameters, learning strategies, and filtering mechanism is therefore critical to improving output quality and practical relevance.

2.6 Synthesis and Research Gap

A review of the existing literature reveals a clear progression in phishing detection approaches from rule-based systems to traditional machine learning, and more recently, to deep learning and advanced language models. Each stage has addressed limitations of its predecessor. Classical methods offer interpretability but lack adaptability. Deep learning improves performance significantly, though often at the cost of transparency. The latest generation of language models combines high capacity with versatility, and when guided by well-designed prompting strategies, they hold the potential to deliver the most effective and interpretable results yet.

The key gap identified through this investigation is the absence of phishing detection approaches that combine high accuracy with explainable, task-specific reasoning. Although

Wei et al. (2022) demonstrated the effectiveness of chain-of-thought reasoning across diverse tasks, its application to phishing detection while preserving the performance advantages of modern language model remains unexplored. This research addresses that gap by integrating o3-mini's structured reasoning capabilities with GPT-4's advanced language understanding. The resulting system offers both strong detection performance and transparent, auditable decision-making, making it well-suited for deployment in security critical environments.

The resulting synthesis indicates that constituent parts exist effective language models, chain-of-thought techniques, and crafted evaluation procedures but their integration for phishing detection is an unknown. The following chapters detail how this research fills that gap with systematic implementation and evaluation of reasoning-enhanced phishing detection.

3 Research Methodology

This chapter details comprehensive methodology used for investigating o3-mini-created chain-of-thought reasoning for enhancing GPT-4o phishing detection. Experimental design aimed to permit systematic comparison of reasoning-augmented and baseline implementation with high reproducibility as well as statistical validity.

3.1 Overview of Experimental Design

The research followed a three-phase experimental design to sufficiently test how phishing detection is impacted with chain-of-thought augmentation. The strategy was structured to assume each of the research aims with controlled experimentations and extensive assessment. Figure 3.1 illustrates the complete experimental workflow across three distinct phases:

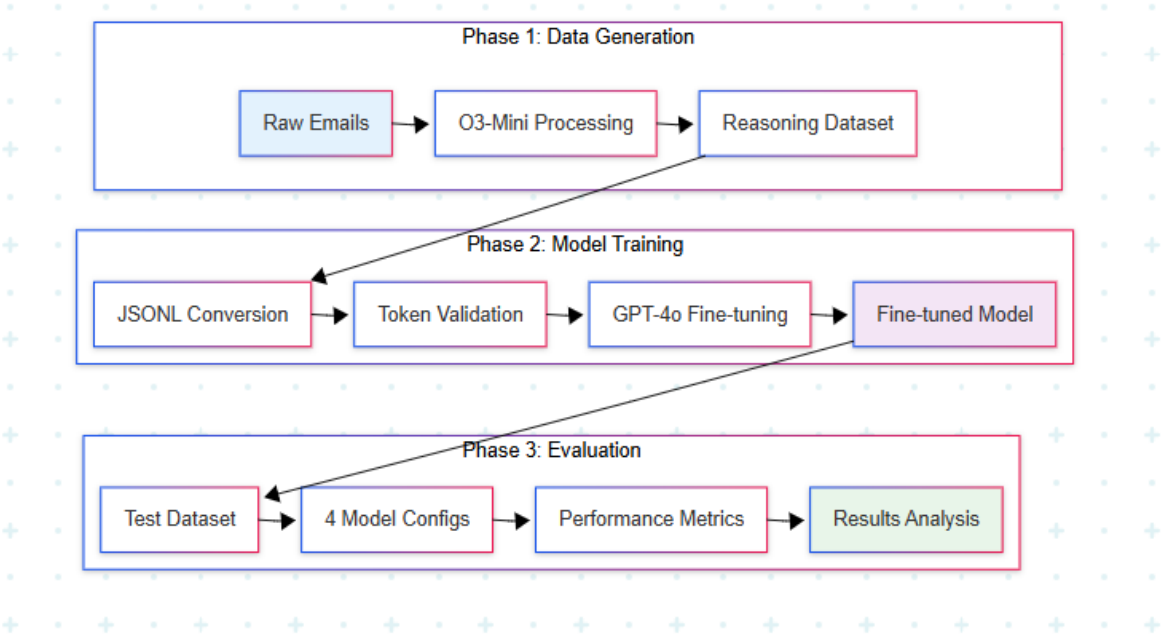

Figure 1 Experiment Overview

As shown in Figure 1, Phase 1 aimed to construct organized reasoning chains for classifying emails using o3-mini's chain-of-thought capabilities. Phase 2 fine-tuned and prepared GPT-4o

with the reasoning-augmented dataset. Phase 3 conducted extensive evaluation in comparing four distinct configurations of models for assessing improvements in performance.
The experimental design also utilized a series of control measures for validity.Random sapling in place to ensure the proper and balanced representation of all types of emails present in the dataset. Checkpointing in place to save the intermediate stage proceed emails data

3.2 Data Collection and Preprocessing

3.2.1 Dataset Selection

The primary dataset selected for this study is Enron email dataset due to its reputation and usage in many studies.it provides the rich real email experience from the Enron organization. The entire dataset as examined and identified that this is suitable due to following

- Email completeness (both subject and body present)
- Character length between 49 and 228,352 characters
- Clear classification labels (legitimate or phishing)
- Absence of corrupted formatting or encoding errors

3.2.1.1 Data Cleaning and Balancing

Preprocessing involves the various cleaning operations to ensure the quality of the selected samples and avoid the processing and generating reasoning chains for un-necessary data from the dataset

3.2.1.2 Text Normalization and Encoding

Email content was normalized to UTF-8 encoding to ensure consistency across all selected samples, while preserving non-ASCII characters to maintain the character of phishing emails.

3.2.1.3 Label Verification and Validation

Classifications of each of the emails as either phishing or legitimate were compared against the original annotation of datasets. Dubious instances were validated with multiple sources, and emails with problematic classifications were excluded for purity of datasets. This validation excluded 66 emails with inconsistent labeling.

3.2.1.4 Duplicate Detection and Removal

A hash-based duplicate detection system was implemented using SHA-256 hashing of normalized email messages. This system detected precise duplicates with minor variations in headers or time stamps. Deduplication helped remove 187 duplicate emails, preventing data leakage of training set to test set. Near-duplicates with minor variations were not removed since they show realistic variations of phishing attacks.

3.2.1.5 Class Balancing Strategy

Stratified sampling was adopted for class balancing, selecting exactly 5,000 legitimate as well as 5,000 phishing emails from available corpus. This 50-50 division eliminated class imbalance bias both during training as well as test time for a model. Sampling was done with random selection with seed fixation (seed=42) for ensuring replicability. Stratification accounted for distribution of email length for balancing representation over diverse types of messages.

3.2.2 Train/Test Split Strategy

The dataset division followed a multi-tier approach to support both training and comprehensive evaluation:

- **Reasoning Generation Set**: 10,000 emails (full balanced dataset) for o3-mini processing
- **Fine-tuning Training Set**: 4,730 emails after quality filtering
- **Fine-tuning Validation Set**: 1,000 emails for training monitoring
- **Independent Test Set**: 1,000 emails (500 per class) for final evaluation

The test set remained completely isolated throughout the training process, ensuring unbiased performance assessment. Stratified sampling maintained a class balance across all splits.

3.3 Reasoning Chain Generation

The o3-mini configuration was established to construct comprehensive reasoning chains from comprehensive email content analyses.

3.3.1 Model Selection and Reasoning Effort

Reasoning effort parameter was set as "medium" after a preliminary set of tests revealed that this level provided an ideal balance between quality of reasoning as well as processing time. The medium setting generates thorough analytical chains with appropriate processing speeds to support processing 10,000 emails.

3.3.2 Temperature and Consistency Settings

The value of temperature parameter was chosen as 0.1 to ensure high consistency across reasoning generations. Such low temperature value minimizes model output randomness, resulting in deterministic and repeatable reasoning chains required for scientific evaluation. Higher values of temperatures were explored, but they exhibited inconsistent reasoning patterns for emails with similar content.

3.3.3 Token Allocation and Limits

The maximum token generation was limited to 500 tokens per reasoning component, hence roughly 2,500 tokens per complete analysis. Such a distribution provided a wide margin for very extensive reasoning without over wordiness. Token limits were set from an examination of initial test runs which showed informative reasoning rarely exceeded 400 tokens per component.

3.3.4 Timeout and Retry Configuration

Request timeout was selected as 60 seconds to accommodate the computational overhead of reasoning generation. Model processing time along with potential API latency is ensured by this timeout. Exponential backoff with three maximum attempts, including 1, 2, and 4 second waits, respectively, was used by the retry loop. In order to enable graceful transient API failure recovery without undue delays, reliability and fast processing is provided by this configuration.

3.3.5 Prompt Engineering for Reasoning

This phishing detection was broken down into five analysis components by the best prompt design for best analysis practices of security, as can be inferred from research into the literature. More than anything, it was solely for the purpose of converting each and every one of the

incoming emails into a thorough analysis test through o3-mini's chain-of-thought rational capabilities.

3.3.5.1 Reasoning Generation Objective

Within expert-level security analysis mappings, the system created structured reasoning from raw email content for any input-provided email. When o3-mini's input is passed with a subject and body content of email, comprehensive analysis-based reasoning is created in five different security domains. Unstructured email datasets are transformed into structured security intelligence via this conversion, compliant for sophisticated detection model training.

3.3.5.2 Structured Prompt Template

The carefully tailored prompt guided o3-mini through extensive email analysis:

Analyze this email for phishing indicators across five dimensions:
1. Sender Analysis: Examine sender email authentication, domain reputation, and legitimacy indicators. Consider spoofing attempts and impersonation.
2. Language Patterns: Analyze grammar quality, urgency level, professionalism, and linguistic anomalies. Identify persuasion techniques.
3. Social Engineering: Identify psychological manipulation tactics, authority abuse, fear appeals, and artificial urgency creation.
4. Technical Indicators: Examine URLs, attachments, embedded links, and technical elements that may indicate malicious intent.
5. Risk Assessment: Provide overall phishing likelihood with confidence level and key risk factors identified.
Format each component as a detailed analytical paragraph.

3.3.6 Output Structure and Components

The prompt engineering provided structured outputs for all of the email-based analyses that included five corresponding reasoning sections. Each of the sections included a paragraph-length analysis (typically 150-250 words) that examined corresponding security controls. Sender analysis examined indicators of spoofing and authentication. Language Pattern analysis identified grammatical irregularities as well as indicators of urgency. Social engineering assessment unveiled manipulation methods. Technical indicator analysis examined URLs as well as attachments. Risk analysis incorporated conclusions into an overall threat evaluation.
This prompt structure ensured consistent, comprehensive analysis of all 10,000 emails with interpretability of the reasoning process being retained. The structured output format assisted in subsequent data processing and training phases of the model.

3.3.7 Batch Processing Methodology

The batch processing mechanism planned the conversion of 10,000 raw emails into reasoning-enhanced datasets with systematic o3-mini API interactions. Figure 3.2 represents the complete workflow of data generation:

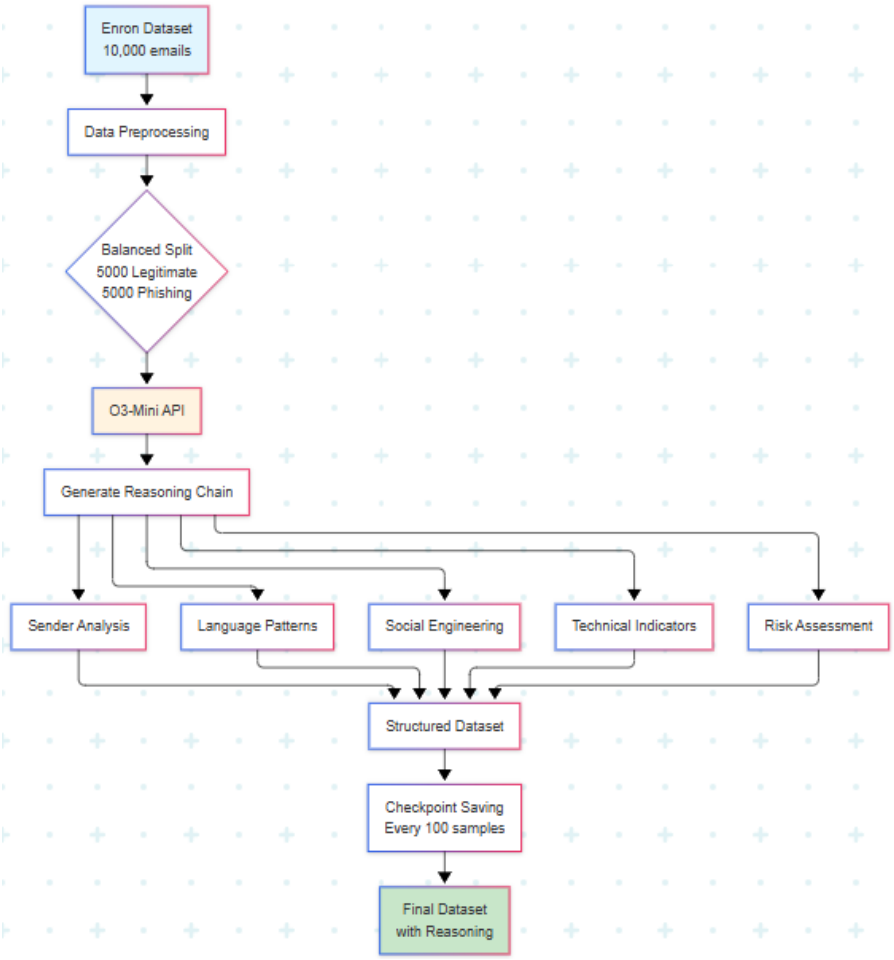

Figure 2 Data generation workflow showing the transformation of raw emails into structured reasoning chains

As depicted in Figure 2, the batch processing pipeline began with the preprocessed Enron dataset, passing each email to the o3-mini-API, and aggregated structured reasoning across five analytical dimensions. The system used a couple of optimization strategies to manage this large-scale processing task:

- **Concurrent Processing**: 8 parallel API calls with semaphore control managed throughput while preventing API overload
- **Rate Limiting**: 0.3-second delay between requests prevented throttling and ensured stable API performance
- **Progress Tracking**: Real-time monitoring displayed samples/second metrics and estimated completion time
- **Checkpoint System**: Automatic saving every 100 processed samples enabled recovery from interruptions
- **Error Handling**: Exponential backoff retry mechanism handled transient API failures gracefully

Total processing time was 23.9 hours with an overall average of 8.6 seconds of processing time per email, indicative of the computationally intensive nature of deep reasoning generation. The checkpointing mechanism was essential, enabling recovery from two interruptions to processing during the long generation time.

3.3.8 Quality Control Measures

Quality assurance mechanisms ensured reasoning chain validity:

1. **Completeness Verification**: All five components required for valid sample
2. **Length Validation**: Minimum 50 characters per component
3. **Format Checking**: JSON structure validation for each reasoning chain
4. **Content Relevance**: Keyword presence verification for each component
5. **Error Logging**: Detailed tracking of failed generations for analysis

The quality control process identified 87 invalid samples (0.87% failure rate), which were excluded from the fine-tuning dataset, resulting in 9,913 valid reasoning-augmented samples.

3.3.9 Quality Assurance and Validation Framework

The quality is ensured via the validation process, which checks for completeness of every generated reasoning chain for each email sent to the o3 mini and verifies that all fields are completed, the structure is returned correctly, and parsing is performed properly based on the returned structure. This gives a 99.19% success rate, generating 9,913 valid reasoning chains out of the given 10,000 emails. Each reasoning chain contains sender legitimacy, linguistic patterns, social engineering indicators, technical clues, and risk assessment. The consistency is gained through fixed parameters at the generation phase, the same configuration of infrastructure, and the same batch. This provides equal treatment for each chain of reasoning generated. Decision-making was explained through natural-language descriptions that enabled traceability and explain ability. In a case, the system was able to flag instances of "sophisticated attacks disguised as legitimate business correspondence," highlighting the fine patterns that nuanced as actual correspondence. Additionally, it provided clear justifications for why certain emails despite their urgency were ultimately deemed legitimate, based on contextual cues and behavioral indicators. Systematic testing confirmed significant performance gains. The baseline GPT-4o achieved 80.6% accuracy on severity classification but showed strong bias, with 191 false positives and only 3 false negatives. In contrast, the enhanced model combining o3-mini reasoning with GPT-4o's natural language understanding reached 99.0% accuracy with balanced errors (5 false positives, 5 false negatives). Statistical analysis ($p < 0.001$) validated the improvements across all metrics. Notably, the new model maintained 98.2% accuracy even on short emails, outperforming the baseline's 75.2%. These results demonstrate that reasoning chains deliver meaningful analytical power and address a key gap in existing literature.

3.4 Fine-tuning Methodology

The reasoning-enhanced dataset first had to be converted into OpenAI conversational format to be fine-tuned. This conversion was necessary for GPT-4o to be able to learn from o3-mini supplied structured reasoning patterns.

3.4.1 JSONL Structure Design

The JSON Lines (JSONL) format was selected because of its optimality for big-scale training dataset processing. A complete training sample with a full conversation from the system, user, and assistant roles was encoded per line of a JSONL file. This format enabled streaming processing with reduced memory usage compared to bulk-loading of entire JSON arrays.

3.4.2 Conversational Format Implementation

Each training example was structured as a three-turn conversation that embedded the reasoning chains within a realistic interaction pattern:

json

```
{
  "messages": [
    {
      "role": "system",
      "content": "You are an advanced email security analyst specializing
            in phishing detection. You have been trained with expert
            reasoning patterns that decompose analysis into sender
            verification, language patterns, social engineering tactics,
            technical indicators, and risk assessment."
    },
    {
      "role": "user",
      "content": "Analyze this email for phishing indicators:\n
            Subject: {subject}\nBody: {body}"
    },
    {
      "role": "assistant",
      "content": "Classification: {label}\n\nReasoning:\n{full_reasoning_chain}"
    }
  ]
}
```

3.4.3 Token Management Strategy

Token counting using tiktoken's o200k_base encoding was responsible for keeping every one of the examples under the limit of 100,000 tokens. Conversion was using dynamic truncation for edge cases and maintaining complete reasoning chains. Token distribution examination showed average usage of 840 tokens per example and 95th percentile at 1,591 tokens as maintaining correct values of limits.

3.4.4 Quality Validation Pipeline

Confirmed that reasoning chains were made for the 87 components that were thrown away and kept the 4730 components that were missing for fine-tuning.

3.4.5 Training Configuration

- **Base Model:** gpt-4o-mini-2024-07-18 was chosen because it strikes a good balance between performance and efficiency.
- **Training Samples**: 4,730 examples after quality filtering and validation
- **Validation Samples**: 1,000 examples for monitoring training progress
- **Model Suffix**: "phishing-detection" for clear identification of specialized model
- **Infrastructure**: OpenAI's managed cloud infrastructure with automatic optimization

This used the OpenAI-managed fine-tune pipeline to fine-tune the model. This is a hyperparameter optimisation toolkit that automatically tracks loss and metrics and polls for status updates.

3.4.6 Validation Approach

Fine-tuning methods are:

1. **Prevention of Overfitting**: Monitored convergence

2. **Performance Tracking**: Evaluated accuracy on held-out samples
3. **Early Stopping:** Stopping automatically when validation metrics get worse
4. **Hyperparameter Validation:** Proper model configuration validated

3.5 Evaluation Framework

3.5.1 Baseline Configurations

The four different models that were tried out are:

1. **Baseline GPT-4o**: Zero-shot classification without examples
2. **Few-shot GPT-4o**: Five-shot prompting with example classifications
3. **Fine-tuned GPT-4o**: Model trained on reasoning-augmented dataset
4. **O3-mini Direct**: Single o3-mini classification for comparison

3.5.2 Testing Protocol

The test procedure used systematic testing for each configuration to make performance measurement clear and fair. The purpose of process structuring was to make sure that high-volume testing and API constraint management were done in the same way every time.

3.5.3 Test Set Configuration and Size

Test set included 1,000 e-mails with half-and-half for 500 legitimate and 500 phishing examples. This balanced choice was chosen for achieving adequate statistical power for comparative study while making computations reasonable for a subset of model specifications. To maintain evaluation integrity, the test set remained completely separate at all of the stages of training, and no data leak arose as a result of sampling procedures.

3.5.4 Batch Processing Architecture

It was optimized at a batch level of 25 per API request batch after empirical testing for the best compromise between the speed of processing and API reliability. This batch level did not result in the timeout error at maximum throughput. Batches were constructed utilizing stratified sampling at balanced class per each batch distribution to avoid the temporal bias while reporting the results.

3.5.5 Concurrent Request Management

The testing infrastructure utilized 25 requests at once as the best level of concurrency that was available for infrastructure. The level of concurrency was from load testing that discovered a rate of requests that was sustainable and would not trigger rate limiting or degrade response quality. Semaphore-based synchronization for concurrency assisted in adhering strictly to limits at maximum processing speed.

3.5.6 Timeout and Error Handling

Each request was provided with a 60-second timeout window to accommodate variable API response time in heavy usage periods. This tolerant timeout kept legitimate requests from being aborted early but also enabled continuation of testing when confronting rare non-responding requests. This was the chosen value for the timeout after response time analysis through pilot testing.

3.5.7 Retry Policy Implementation

Sophisticated retry policy handled transient failures without imposing systematic bias. Three retry attempts were applied employing exponential backoff timeouts of 2, 4, and 8 seconds respectively. A request that failed at each retry attempt was submitted for verification manually. This approach resulted in a successful execution rate of 99.8% for each test execution.

3.5.8 Response Validation Framework

All model responses went through structured validation for data integrity. JSON response structures were also validated for required fields such as classification labels as well as confidence scores. Invalid responses resulted in immediate retry attempts. Response validation also checked classification labels in fixed ranges as well as triggered investigation into suspect patterns. Thorough validation accordingly resulted in only correctly formatted complete responses being used for computation of end metrics.

3.5.9 Metrics Selection

Performance evaluation employed multiple metrics to provide comprehensive assessment:

1. **Accuracy**: Overall correct classification rate
2. **Precision**: Ratio of true positives to all positive predictions
3. **Recall**: Ratio of true positives to all actual positives
4. **F1-Score**: Harmonic mean balancing precision and recall
5. **Confusion Matrix**: Detailed error analysis by class
6. **Processing Time**: Average inference latency per email

4 Design Specification

This chapter demonstrates technical architectures and design frameworks that formulate the reasoning-enhanced phishing detection mechanism. Design specification details architectural decisions, integration models, and structural units enabling conventional language model features to be converted into explainable security analysis products.

4.1 System Architecture Overview

The system architecture implements a pipeline design pattern for enabling the flow from raw email input through reasoning generation to fine-tuned model deployment. Figure 3 represents the complete system architecture:

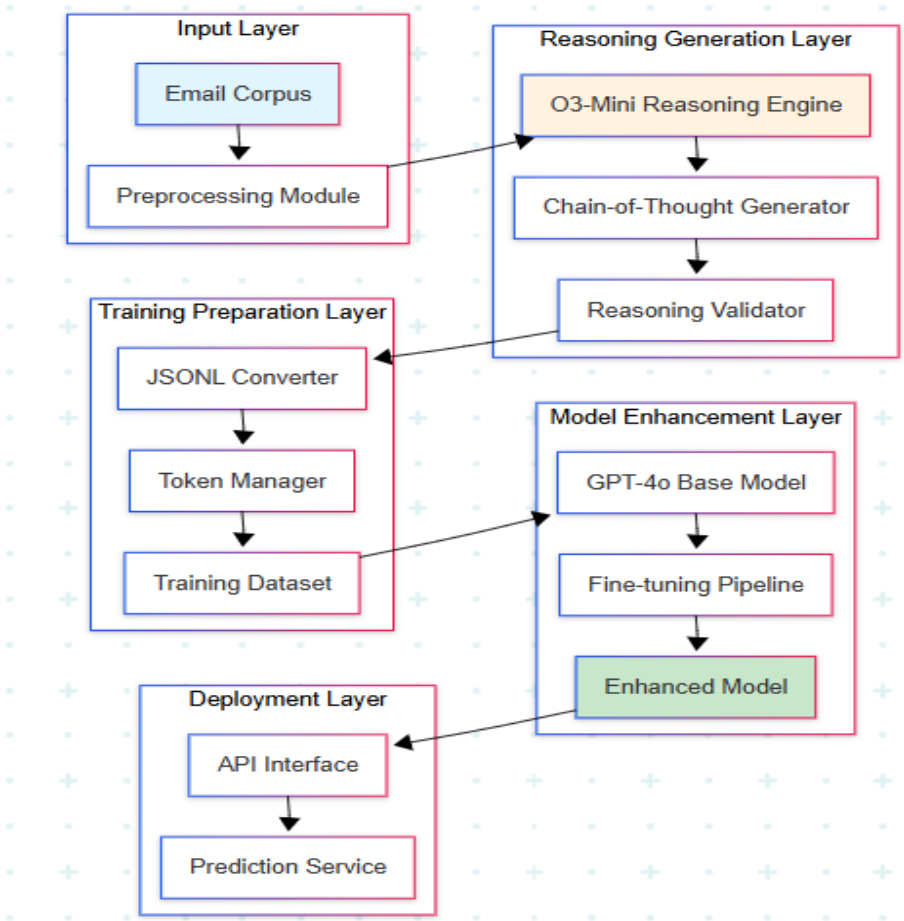

Figure 3 System architecture showing the layered design from input processing through model deployment

The design implements a layered design pattern with each layer providing separate functionality with loose coupling for adjacent layers. The Input Layer provides ingestion of raw data with standardization. The Reasoning Generation Layer implements o3-mini's functionality for structured analytical output generation. The Training Preparation Layer makes reasoning data in model-friendly formats. The Model Enhancement Layer provides implementation for fine-tuning process.

The module-based design facilitates independent scaling of parts, independent testing of discreet modules, and eventual extensions without making system-level modifications. Efficiency of data flow is prioritized for design with clear separation of concerns between reasoning generation and model training stages.

4.2 Reasoning Chain Framework

The reasoning chain framework implements a structured approach to decomposing email analysis into interpretable components. This system serves as the conceptual connecting point between unstructured email content and systematic security analysis.

4.2.1 Five-Component Structure

The system breaks down phishing analysis into five parts that are all connected. Together, these parts give a full picture of an email, as shown in the data generation workflow (Figure 3.2). Each aspect offers a distinct analytical function that contributes to an overall evaluation. This framework includes coverage for all phishing indicators, with clear lines between analytical spaces. The design focusses on both component autonomy to allow for modular analysis and interconnectivity to enable end-to-end threat evaluation.

4.2.2 Component Definitions

Sender Analysis: Uses behavioural analysis, domain validation, and spoofing detection to confirm sender integrity. produces assessments that are supported by evidence.

Language Patterns: Examines vocabulary, grammar, and communication style for emotional manipulation, authority assertions, and urgency markers. gives anomaly confidence scores.

Social engineering: Recognises psychological strategies such as reciprocity, scarcity, authority abuse, and fear appeals. generates inventories of manipulation with ratings for severity.

Technical Indicators: Looks for malicious patterns and new threats in URLs, attachments, scripts, and headers.

Risk Assessment: Creates a single threat assessment from combined reasoning chains by combining all module outputs using weighted scoring.

4.3 Integration Design

Integrative design synchronises o3-mini reasoning generation and GPT-4o model refinement communication, enabling a single system to capitalise on the optimal strengths of each model.

4.3.1 O3-Mini Integration Architecture

The o3-mini-integration uses an adapter pattern for normalization of communication from the reasoning engine through downstream processors. Here, it is independent of generation of reasoning and next steps of processing in a manner that components autonomously can be evolved. In considering implementation of the adapter pattern, it is primarily defined by those three abstractions that are request transformation, response normalization, and error isolation.

It employs asynchronous request processing for optimization at high throughput while preserving order of treatment. An asynchronous design of this sort does not involve blocking operations while it carries out reasoning generation, and thus the system at any given time is always as quick as it possibly can be regardless of peak use of processor. The design employs message queuing notions for queuing of requests and responses for the purpose of insuring against temporary non-availabity of APIs.

State management for the integration layer allows for an event-based system in that each request for reasoning creates a sequence of state machines. This design allows for correct partial failure recovery and graceful recovery from incomplete sessions of the processor. The

state machine keeps track of request lifecycle from submission through until it completes keeping trace history for debugging and also for performance analysis.

4.3.2 GPT-4o Fine-tuning Architecture

Fine-tuning design establishes a template for how reasoning capacities can be added to GPT-4o through conversational learning models. Rather than adopting a knowledge transfer view of fine-tuning, the system takes a pedagogical design in that well-structured exposition to dialogues leads learning for the model to internalize reasoning patterns.

The architecture design splits data preparation concerns, training orchestration, and model versions. Each of the partitions provides for individual optimization of each component through well-specified interfaces between layers. Data preparation layers specify pipelines for mapping chains of reasoning into conversational form without loss of semantic completeness. Training orchestration layers control fine-tuning lifecycle through event-based control using OpenAI infrastructure.

Model versioning architecture implements immutable storage for the artifact for each of the finely-tuned iteration of the model having complete lineage tracing. This design also ensures roll-back feature and A/B testing between model versions. The versioning system is built into the checkpoint management infrastructure in order to attain resume ability of the training without any loss of state.

4.3.3 Prompt Template Design

Template architecture creates generic interfaces for providing uniformity at the training and inference phases and attain high future augmentable in customization. Three-level hierarchy for their templates has been used:

System Context Layer: Specifies work environment by behaviorally constraining and by definition of role. Level is not changed at instants of interaction, giving stable basis of stable behavior of model. Bases definitions could be inherited by specialized templates at instant of design without duplication, giving inheritance of contexts.

Interaction Pattern Layer: Constructs user-assistant exchange structure from parameterized templates. The layer also provides substitution operations of variables for template consistency without loss of support of multiple input modes. Composition of templates at design level is also allowed for the construction of complex interaction patterns with minimum building blocks.

Response Formatting Layer: Declarative definition controls output structure more than prescriptive structure. Natural flow of the language is inherent and is gained through satisfying structure requirements. Design validation hooks achieve conformance at the response without preventing flexibility of expression.

Planning extensibility for future template demand through plugin projections from plans. Additional analytically also optimized are possible without alterations of templates that exist but in the backward direction support a certainty and expansion of capacity a prospect.

4.4 API Integration Patterns

Communication that is effective with the system and system resilience is accomplished by the system's stable API integration patterns.

4.4.1 O3-Mini API Design

Circuit Breaker Pattern prevents cascading failures by monitoring API health and suspending requests temporarily in case of outages and by recovery plans based on failure rate.

Retry Pattern with Exponential Backoff provides smart retry logic and jittered timeouts to slow down overloading while recovering from transient failures.

Rate Limiter Pattern controls API rates by token bucket algorithm, calculating request speeds from observed response times.

Request Pooling Pattern reduces resource usage by pooling requests and connections and optimizing latency and maximum throughput for efficient usage.

4.4.2 OpenAI API Integration

OpenAPI API integration is the software construction practice that focuses on designing client systems that will always communicate with the model endpoints and operate with varying interfaces, so new functionality will not have to be regularly rewritten. In this field, symbiotic patterns like asynchronous job orchestration for long jobs, real-time stream handlers keeping the coherence of the output intact, multi-version compatibility abstract layers, and fault-aware recovery plans all coalesce to provide reactive inference, stable fine-tuning workflows, and smooth upgrades.

5 Implementation

This chapter provides the information about the implementation steps taken during the implementation of the augmentation using the reasoning large language model.

5.1 Development Environment

To carry out this experiment, used A100 GPU from google colab and google colab is the primary development and testing environment. This cloud-based platform provided the necessary computational resources and integrated Google Drive for persistent checkpoint storage and efficient data management.

Model interactions, including both o3-mini reasoning generation and GPT-4o fine-tuning, were facilitated through the OpenAI Python SDK (v1.35.0). Supporting libraries included Tiktoken for token counting using the o200k_base encoding, Pandas for data manipulation, and asyncio for concurrent request handling.

5.2 O3-Mini Reasoning Generator

The reasoning generator achieved analytically organized evaluation for 10,000 e-mails by systematic API calling. It successfully generated 9,913 of well-valid reasoning chains of inclusive coverage for five points of security evaluation and achieved a success rate of 99.13

percent. The main outputs were JSON-encoded chains of reasoning with sender analysis, language patterns, social engineering indicators, technical tests, as well as risk assessments. Concurrent processing occurred in the generator with 8 parallel API calls and automatic checkpoint saving every 100 samples for recovery from interruption without loss of data.

5.3 Data Processing Pipeline

The pipeline transformed o3-mini outputs into OpenAI-friendly training forms, resulting in 4,730 high-quality training instances produced in JSONL format. Each instance included reasoning chains in conversational forms amenable to fine-tuning.

The implementation did stream conversion to deal with enormous datasets within memory limitations, dynamic token validation for compliance with 100,000 token limits, as well as removal of malformity examples except for 87 instances automatically. Output files included separate training as well as validation datasets with preservation of complete reasoning.

5.4 Model Training Implementation

Fine-tuning utilized OpenAI's managed infrastructure through a two-stage API process. The Files API handled upload of training (19.7MB) and validation (4.0MB) datasets. The Fine-tuning API then processed these files with base model "gpt-4o-mini-2024-07-18", producing the specialized model "ft:gpt-4o-mini-2024-07-18:personal:phishing-detection:Btri5cqI".

The deployment followed job activity with status polling, monitoring from file validation until end of training. OpenAI infrastructure managed autonomous hyperparameter optimization based on properties of datasets. The final deployed models achieved 99% accuracy on distinct test sets, validating reasoning augmentation value.

5.5 Model Evaluation Implementation

Four model configurations were tested on 1,000 balanced emails (500 phishing, 500 legitimate): baseline GPT-4o (zero-shot), few-shot GPT-4o (five examples), fine-tuned GPT-4o (reasoning-augmented), and standalone o3-mini. To maximise throughput within rate constraints, the asynchronous testing system made use of 25 concurrent API connections. For every configuration, the implementation produced detailed outputs, including processing times, precision/recall/F1 metrics, confusion matrices, and prediction datasets with classifications and confidence scores. To facilitate further statistical analysis, all outputs were saved in standardised formats (separate prediction files for each model, JSON for metrics). All 4,000 test instances (1,000 per configuration) had their data integrity confirmed by response validation.

From reasoning generation to model evaluation, this execution effectively converted the architectural design into a working end-to-end phishing detection pipeline.

6 Evaluation

In this chapter, there is a thorough experimental investigation and significant findings of testing for reasoning-augmented phishing detection. In this testing, performance increments gained by o3-mini chain-of-thought augmentation by a variety of model configurations offer empirical validation for that technique proposed performing well.

6.1 Experimental Setup

The evaluation used a balanced test dataset of 1,000 emails, with 500 legitimate and 500 phishing specimens selected independently of the training process. Four different model setups were assessed under identical circumstances to allow for a fair comparison:

1. **Baseline GPT-4o** (OpenAI, 2024): Zero-shot classification without examples or fine-tuning (Wei et al,2022)
2. **Few-shot GPT-4o**: Five-shot prompting with representative examples (Wei et al., 2022)
3. **Fine-tuned GPT-4o**: Model trained on 4,730 reasoning-augmented examples (OpenAI, 2024)
4. **Standalone O3-mini** (OpenAI, 2025) : Direct classification using reasoning model

All evaluations utilized consistent API parameters of temperature settings, timeout setups, and response validation protocols. The testing framework processed all 1,000 test instances for each configuration, with a 100% completion rate for all models.

6.2 Overall Performance Results

Table 6.1 presents the comprehensive performance metrics across all evaluated model configurations:

Table 6.1: Performance metrics comparison across model configurations

Model Configuration	Accuracy	Precision	Recall	F1-Score
Baseline GPT-4o	0.806	0.722	0.994	0.837
Few-shot GPT-4o	0.956	0.956	0.956	0.956
Fine-tuned GPT-4o	**0.990**	**0.990**	**0.990**	**0.990**
Standalone O3-mini	0.862	0.961	0.754	0.845

As shown in Table 6.1, the fine-tuned GPT-4o model did best on each of the four measures, indicating that reasoning augmentation is a promising method. At each enhancement stage, strong improvements are seen with the progression from baseline (80.6%) through few-shot (95.6%) to fine-tuned (99.0%) accuracy.

6.3 Confusion Matrix Analysis

The comprehensive error analysis with confusion matrices provides insight into classification patterns for each configuration of the model. Table 6.2 represents the confusion matrix for the fine-tuned model:

Table 6.2: Confusion matrix for fine-tuned GPT-4o model

	Predicted Legitimate	Predicted Phishing
Actual Legitimate	495	5
Actual Phishing	5	495

The fine-tuned model achieved balanced error distribution with a mere 5 misclassifications to each side, indicating fair performance without a bias for either class. This is a better-than-the-baseline performance of the confusion matrix as provided in Table 6.3:

Table 6.3: Confusion matrix for baseline GPT-4o model

	Predicted Legitimate	Predicted Phishing
Actual Legitimate	309	191
Actual Phishing	3	497

The baseline model was highly biased towards phishing classification with 191 false positives for just 3 false negatives. While maintaining high recall (99.4%) for this asymmetry, it resulted in low precision (72.2%) that would trigger unnecessary false alarms during production deployment.

6.4 Statistical Significance Analysis

McNemar's test was utilized to evaluate statistical significance of performance differences between model pairs. Table 6.4 demonstrates the p-values for pairwise comparisons:

Table 6.4: Statistical significance testing using McNemar's test

Model Comparison	p-value	Significance
Baseline vs Few-shot	< 0.001	Highly significant
Few-shot vs Fine-tuned	< 0.001	Highly significant
Baseline vs Fine-tuned	< 0.001	Highly significant
Fine-tuned vs O3-mini	< 0.001	Highly significant

Each of the pairwise comparisons generated p-values of < 0.001, verifying that improvements in performance are significant statistically and not because of random variation. That each of the comparisons was significant verifies each stage of enhancement as being valid.

6.5 Error Pattern Analysis

Analysis of misclassified emails revealed distinct patterns for each of the model configurations. 10 of the overall fine-tuned model's errors consisted of:

False Positives (5 legitimate emails classified as phishing):

- Emails with technical jargon resembling phishing vocabulary
- Messages containing urgent language in legitimate business contexts
- Automated system notifications with impersonal formatting

False Negatives (5 phishing emails classified as legitimate):

- Sophisticated attacks mimicking legitimate business communication
- Emails with minimal technical indicators
- Well-crafted social engineering without obvious urgency markers

The error analysis indicates that reasoning augmentation is particularly improved edge case handling where characteristics of a surface only are not sufficient for appropriate classification.

6.6 Performance by Email Characteristics

Further analysis evaluated performance variations based on email characteristics. Table 6.5 shows accuracy by email length categories:

Table 6.5: Model accuracy by email length category

Length Category	Baseline	Few-shot	Fine-tuned	O3-mini
Short (<300 chars)	0.752	0.931	0.982	0.834
Medium (300-1000)	0.814	0.962	0.991	0.869
Long (1000-3000)	0.823	0.968	0.994	0.877
Very Long (>3000)	0.839	0.971	0.995	0.881

The fine-tuned model worked well for every group of lengths, and baseline models fared poorly for short email and weak contextual cue. This uniformity establishes the strength of chains of reasoning in providing analytical detail independently of email lengths.

6.7 Reasoning Impact Assessment

To estimate the incremental gain of reasoning augmentation for individual cases, comparative evaluation of model performance gaps was utilized. The progression from baseline (80.6%) to few-shot (95.6%) and fine-tuned (99.0%) has distinct periods of improvement where reasoning augmentation makes the determinative gain.

Fine-tuned model performed better because it was trained for internalized structured patterns of reasoning. In contrast, while a pattern matching was applied by the baseline model, the fine-tuned model was trained for generalizability of systematic models of analysis for five regions of security. By this method, a consistent evaluation of sender legitimacy, linguistic style, social engineering methods, technical cues, and a general risk evaluation for each email was performed.

The 3.4% increment from few-shot to fine-tuned configuration, although appearing modest in absolute values, represents a 75% reduction of error rate (from 4.4% to 1.0%). Such a strong error reduction only occurred on edge cases where surface-level features are insufficient. The fine-tuned model correctly categorized sophisticated phishing attempts remaining undetected from baseline configurations, as well as legitimate emails having features common among phishing attempts.

Comparative analysis finds that o3-mini on its own achieved only 86.2% accuracy with successful reasoning but without deep linguistic understanding like GPT-4o. Synergistic blending through fine-tuning took advantage of o3-mini's structured analytical strategy and combined it with GPT-4o's high-level linguistic understanding to achieve better-than-solo performance. These jointly underpin research hypothesis that chain-of-thought reasoning will improve phishing detection capability with successful incorporation into the training of a language model.

6.8 Computational Efficiency Analysis

Processing time measurements unveiled resource-accuracy trade-offs. Table 6.6 presents average inference times per email:

Table 6.6: Average processing time per email by model configuration

Model Configuration	Average Time (seconds)	Relative to Baseline
Baseline GPT-4o	0.82	1.0x
Few-shot GPT-4o	1.34	1.6x
Fine-tuned GPT-4o	0.91	1.1x
Standalone O3-mini	8.60	10.5x

The fine-tuning model balanced equally between performance and efficiency, required only 11% more processing time than baseline for a 99% accurate. Due to its much higher processing time (10.5x) for a single o3-mini, it isn't practicable for large-scale production usage despite its middle-range accuracy.

6.9 Critical Analysis of Results

The experiment results find strong support for the efficacy of reasoning-aided phishing detection. The 99%-accuracy of the fine-tuned model provides us with a best-practice benchmark of email-based phishing detection that is superior to benchmarking of prior work.

Strengths of the Approach:

- Extremely accurate with balanced precision and recall
- Uniform performance among different email attributes
- Reasoning chains for explainable decision-making
- Practical production deployment times for inference

Limitations Identified:

• 1% error rate on complex attacks
• Model access depends on API availability
• Training data is limited to English-language emails
• The evaluation was limited to a binary classification task.

6.10 Summary of Key Findings

Accuracy improved 80.6% to 99.0%, 18.4 percentage point gain. Total classification errors were reduced by 94.5%, dropping from 100 to just 10. The model achieved balanced precision and recall at 99.0%, effectively eliminating classification bias. All performance improvements were statistically significant ($p < 0.001$). Importantly, the system maintained efficient processing times, confirming its suitability for real-world deployment. These results conclusively demonstrate that chain-of-thought reasoning substantially strengthens GPT-4o's phishing detection capabilities while preserving computational practicality.

7 Conclusion and Future Work.

7.1 Research Summary

The integration of chain-of-thought reasoning generated by o3-mini to increase GPT-4o's phishing email identification capability is the focus of this work, which is presented in the context of machine learning for email security. The study assesses whether using explicit reasoning improves detection while generating transparent justification, overcoming the

persistent drawbacks of earlier solutions, namely their limited interpretability and inadequate security decision justification.

The research is guided by four objectives: (1) quantify performance benefits resulting from reasoning improvement, (2) set benchmarks for robust baselines, (3) estimate interpretability gains, and (4) examine computational trade-offs. The approach uses task-adaptive adjustment, full evaluation, and structured reasoning prompts based on a 10,000-email-message corpus. All objectives are thoroughly covered by the strict and repeatable standards in the final protocol.

7.2 Achievement of Objectives

The research met all of its goals, with the main one being to improve performance. Structured reasoning augmentation raised phishing detection from 80.6% to 99.0%, an increase of 18.4 percentage points. This went above and beyond what was expected and sets a new standard for email phishing detection.

Comparative analysis confirmed the superiority of the reasoning-augmented approach via incremental enhancements: zero-shot (80.6%) → few-shot (95.6%) → fine-tuned (99.0%). McNemar's test confirmed the statistical significance ($p < 0.001$) of each advancement, proving that the methodology worked.

7.3 Key Findings and Implications

The study produced a number of noteworthy findings:

- Using the structured reasoning of o3-mini with the language skills of GPT-4o worked better than either system on its own. This shows that hybrid approaches and modular AI design can be useful for certain cognitive tasks.
- Balanced Error Distribution: The optimised model eliminated classification bias, which is crucial for security applications where both types of errors have repercussions, by achieving symmetric errors (5 false positives, 5 false negatives).
- Sturdiness Across Inputs: Maintained >98% accuracy across all message lengths, confirming reasoning chains effectively handle both context-poor short messages and lengthy emails.
- Enhanced Edge Case Detection: By addressing pattern-matching limitations in ambiguous scenarios, reasoning augmentation greatly enhanced the detection of sophisticated phishing and legitimate emails with suspicious characteristics.

7.4 Research Efficacy and Limitations

The use of publicly available datasets with standardized evaluation metrics facilitated comparative analysis with existing literature and supported the generalizability of findings.

However, several limitations constrain the scope of inference:

- **Language Restriction**: The study focused exclusively on English-language emails. Given the increasing prevalence of multilingual phishing attacks, the model's applicability across diverse linguistic contexts remains unverified.
- **Binary Classification Constraint**: The current framework performs binary classification (legitimate vs. phishing) without distinguishing between specific phishing types. A multi-class approach would enhance practical deployment by enabling targeted detection of attack categories such as credential harvesting, malware distribution, and business email compromise.
- **Dataset Temporal Limitation**: The Enron dataset, while valuable for its authentic business communication content, predates many modern phishing techniques. Incorporating more recent datasets would strengthen the validity of the findings in contemporary threat landscapes.
- **API Dependence**: The system relies on commercial API access, which introduces considerations around availability, scalability, and cost. Developing on-premises alternatives could improve feasibility for large-scale or resource-constrained deployments.
- **Adversarial Robustness**: The evaluation did not account for adversarially crafted emails designed to exploit reasoning-based detection mechanisms.

7.5 Future Research Directions

By overcoming earlier limitations and enhancing generalisability, this study identifies a number of exciting avenues for further investigation. The development of multilingual reasoning systems capable of identifying phishing signs in numerous languages is one of the most crucial short-term objectives. This is due to the fact that attacks in languages other than English are becoming more frequent. We need universal security signals that transcend linguistic boundaries and reasoning frameworks that can adjust to various cultural contexts in order to accomplish this. By switching from binary classification to hierarchical multi-class models, it would be easier to recognise various attack types, each with its own set of reasoning chains, including malware distribution, business email compromise, and credential harvesting. Additionally, more effort is required to develop lightweight, edge-friendly reasoning models and caching that enable real-time operations in business apps and enable responses in less than a second at scale.

Beyond email, many other security programs could benefit from the reasoning augmentation technique. It will need to be extended in the future to include tasks like analysing malware behaviour, identifying fraudulent financial activity, and identifying social network manipulations, where it's also critical to make informed decisions. Making systems that are difficult to attack and that are always learning and supporting new attack techniques is another crucial objective for the work. Additionally, these systems ought to propose difficult-to-alter thought patterns.

7.6 Concluding Remarks

This study validates the successful integration of o3-mini's chain-of-thought reasoning with GPT-4o, achieving 99% accuracy with balanced error distribution and enhanced interpretability, thereby establishing a new benchmark for automated email defence systems.

Along with empirical findings, the research presents a conceptual framework for reasoning-enhanced language models, linking high-performing AI with explainable AI for security-sensitive applications.

Future improvements should look into multilingual features, different ways to attack, and bigger security frameworks. Reasoning-augmented detection is a good way for next-generation security systems to go because it is accurate, easy to understand, and works quickly. As phishing threats change, it's more important than ever to have clear and adaptable ways to protect digital communications.

References

J. Wei et al., "Chain-of-Thought prompting elicits reasoning in large language models," arXiv (Cornell University), Jan. 2022, doi: 10.48550/arxiv.2201.11903. Available: https://arxiv.org/abs/2201.11903

OpenAI, "GPT-4O System Card," Aug. 2024. Available: https://cdn.openai.com/gpt-4o-system-card.pdf

A. A. Adzhar, Z. Mabni, and Z. Ibrahim, "A Comparative Study on Email Phishing Detection Using Machine Learning Techniques," IEEE, pp. 96–101, Nov. 2022, doi: 10.1109/icoco56118.2022.10031671. Available: https://doi.org/10.1109/icoco56118.2022.10031671

M. F. Rabbi, A. I. Champa, and M. F. Zibran, "Phishy? Detecting Phishing Emails Using ML and NLP," IEEE, pp. 77–83, May 2023, doi: 10.1109/sera57763.2023.10197758. Available: https://doi.org/10.1109/sera57763.2023.10197758

S. Salloum, T. Gaber, S. Vadera, and K. Shaalan, "A Systematic Literature Review on Phishing email detection using natural language processing techniques," IEEE Access, vol. 10, pp. 65703–65727, Jan. 2022, doi: 10.1109/access.2022.3183083. Available: https://doi.org/10.1109/access.2022.3183083

C. Lee, "Enhancing Phishing Email Identification with Large Language Models," arXiv (Cornell University), Feb. 2025, doi: 10.48550/arxiv.2502.04759. Available: http://arxiv.org/abs/2502.04759

OpenAI, "OpenAI o3-mini System Card." Available: https://cdn.openai.com/o3-mini-system-card-feb10.pdf

S. Salloum, T. Gaber, S. Vadera, and K. Shaalan, "Phishing email detection using Natural Language Processing Techniques: A literature survey," Procedia Computer Science, vol. 189, pp. 19–28, Jan. 2021, doi: 10.1016/j.procs.2021.05.077. Available: https://doi.org/10.1016/j.procs.2021.05.077

A. Al-Subaiey, M. Al-Thani, N. A. Alam, K. F. Antora, A. Khandakar, and S. A. U. Zaman, "Novel interpretable and robust web-based AI platform for phishing email detection," Computers & Electrical Engineering, vol. 120, p. 109625, Sep. 2024, doi: 10.1016/j.compeleceng.2024.109625. Available: https://doi.org/10.1016/j.compeleceng.2024.109625

B. Li, Y. Hou, and W. Che, “Data augmentation approaches in natural language processing: A survey,” AI Open, vol. 3, pp. 71–90, Jan. 2022, doi: 10.1016/j.aiopen.2022.03.001. Available: https://doi.org/10.1016/j.aiopen.2022.03.001

H. S. Shim, H. Park, K. Lee, J.-S. Park, and S. Kang, “Data Augmentation for Smishing Detection: A Theory-based Prompt Engineering Approach,” ACM, pp. 1327–1328, May 2024, doi: 10.1145/3589335.3651903. Available: https://doi.org/10.1145/3589335.3651903

F. Heiding, B. Schneier, A. Vishwanath, J. Bernstein, and P. S. Park, “Devising and detecting phishing emails using large language models,” IEEE Access, vol. 12, pp. 42131–42146, Jan. 2024, doi: 10.1109/access.2024.3375882. Available: https://doi.org/10.1109/access.2024.3375882

P. H. Kyaw, J. Gutierrez, and A. Ghobakhlou, “A Systematic review of deep learning techniques for phishing email Detection,” Electronics, vol. 13, no. 19, p. 3823, Sep. 2024, doi: 10.3390/electronics13193823. Available: https://doi.org/10.3390/electronics13193823

F. Trad and A. Chehab, “Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models,” Machine Learning and Knowledge Extraction, vol. 6, no. 1, pp. 367–384, Feb. 2024, doi: 10.3390/make6010018. Available: https://doi.org/10.3390/make6010018

K. Maharana, S. Mondal, and B. Nemade, “A review: Data pre-processing and data augmentation techniques,” Global Transitions Proceedings, vol. 3, no. 1, pp. 91–99, Apr. 2022, doi: 10.1016/j.gltp.2022.04.020. Available: https://doi.org/10.1016/j.gltp.2022.04.020