PHISHING EMAIL DETECTION SYSTEM USING BERT

Lee Martins
Student# 1008866835
leah.martins@mail.utoronto.ca

Asmita Chandra
Student# 1005678901
asmita.chandra@mail.utoronto.ca

Riya Kapoor
Student# 1009030558
riya.kapoor@mail.utoronto.ca

Joonseo Park
Student# 1008819281
joonseo.park@mail.utoronto.ca

ABSTRACT

This document outlines our plan to develop a deep learning model that acts as a Phishing Email Detection tool. The general framework utilizes a **Bidirectional Encoder Representation from Transformers (BERT)** model. Throughout the document there is more in depth information regarding data selection, alternative models, team deliverables and the overall architecture for this project to succeed. – Total Pages: 9

1 INTRODUCTION

As personal and professional interactions continue to migrate online, cyberattacks in the form of phishing emails pose an ever-growing threat. As these cyberattacks evolve, so do their ability to bypass the built-in security features of email platforms, resulting in a growing need for intelligent systems that detect subtle patterns seen in phishing emails.

The primary goal of our project is to develop a deep-learning based system capable of differentiating between legitimate emails and deceptive "phishing" emails with minimal error. This project directly addresses real-world cybersecurity threats with potential applications for individual users and corporate IT infrastructures. With the growing sophistication, ever-evolving nature, and frequency of phishing emails, the importance of our project becomes apparent.

A deep learning model is well suited, as it can recognize subtle cues in email content while continuously learning from the patterns present in the latest phishing methods. In contrast, traditional rule-based detection systems fail to recognize these subtle linguistic details. This is more than a project, but moreover, a powerful tool to enhance digital security.

2 ILLUSTRATION

Figure 1 below illustrates the overall architecture of our email phishing detection project.

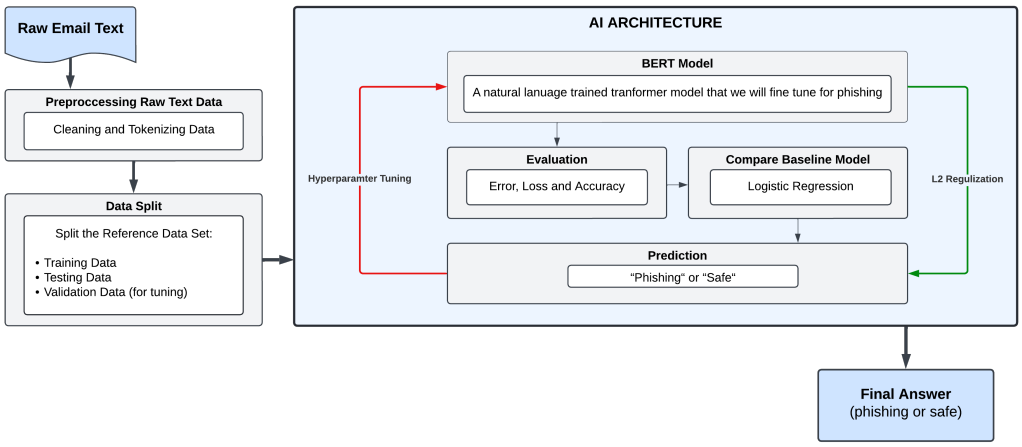

Figure 1: Explicit diagram of our AI model highlighting the BERT model

The diagram outlines the main stages of our workflow and shows our end-to-end process from the original email to the model's phishing prediction. Note that further details of each stage will be provided throughout this document.

3 BACKGROUND AND RELATED WORK

To provide context to our phishing email detection system, we reviewed several prior works in the email threat detection domain. The following are related works:

1. **CNN-Based Phishing Detection:** A common model for phishing detection is a Convolutional Neural Network. This method takes features from email content using CNN layers and has shown effectiveness in binary classification tasks. The IEEE paper Kumar et al. (2024) emphasizes how CNN is used to detect phishing emails but learning text-based patterns in datasets. Even though CNNs are faster to train, they perform worse than transformer models like BERT in detecting language dependencies. This study also provides insight into which regularization models are more effective over others. This has been taken into consideration for our project.
2. **How to use BERT - Visual Guide:** This interactive visual notebook Alammar (2020) forms the foundation for our understanding of how to tokenize, encode and fine-tune BERT for text classification tasks. It guides our model development process and architecture decisions, especially for embedding layers and classification token usage.
3. **LLaMa-PhishSense-1B:** The Llama-Phishsense-1B model AcuteShrewdSecurity (2024) is an open source phishing detector based on Casual Language Model with LoRA-adapted fine-tuning. The model demonstrates the practicality of adapting general-purpose language models for cybersecurity tasks and shows how emerging LoRA also known as Low-Rank Adaption techniques can reduce compute cost during fine-tuning.
4. **Machine Learning Approach for Email Phishing Detection:** This study Al-Shabi et al. (2024) presented a machine learning pipeline for phishing email detection focusing on extracting text-based features from email headers and bodies. Several classifiers are evaluated using logistic regression and support vector machine (SVM). The work highlights the importance of feature selection and pre-processing, showing how combining header and content features significantly enhances phishing detection performance.
5. **A Systemic Literature Review on Phishing Email Detection using Natural Language Processing Techniques:** In this study Alshamrani et al. (2022), the authors extract a

wide range of handcrafted features from both email headers and body text for instance: sender-domain characteristics, link counts, HTML tag usage and message length. The study focuses on how robust feature engineering improves phishing detection accuracy and reduces false positives, demonstrating the value of combining textual and structural signals before applying classifiers.

4 Data Processing

4.1 List of Datasets Used

Table 1: List of Datasets

Dataset	Description
Phish No More Alam (2023)	• All samples are labeled ”legitimate” or “spam/phishing” • Some samples contain more email fields such as sender, recipient, date, and number of URLs, while other samples contain only subject and body • There are 82, 500 emails in total, with 42, 891 “spam” emails and 39, 595 “legitimate” emails
University of Twente Phishing Email Validation Dataset Miltchev et al. (2024)	• All samples are labeled “safe email” or “phishing email” • Dataset contains both human-written and artificially generated emails • Each sample consists of “email text” and “email type” fields • Intended for validating “performance of models after training” [number]
Cyber Cop Phishing Email Detection Cop (2023)	• All samples labeled either “phishing email” or “safe email” • All samples contain an “email text” and “email type” field • Contains 18,000 emails generated by Enron corporation employees
Human-LLM Generated Phishing-Legitimate Emails Greco (2024)	• There are two datasets; one consisting of phishing emails and the other consisting of safe emails • Both datasets (phishing and safe) consist of human-written and LLM generated emails • The LLMs used to generate emails were WormGPT and ChatGPT • Both datasets (phishing and safe) contain 1000 human-generated and 1000 LLM generated emails • The samples consist of a sender, receiver, date, subject, body, URL count, and a label (0 for human generated and 1 for LLM generated)

4.2 Consolidating the Datasets

The datasets in 4.1 should be merged into one, standardizing columns to include: label, body, sender, receiver, date, subject, and URL count. Fields like “email text” from the Twente and Cyber Cop datasets should be renamed to “body”. Cyber Cop labels should be converted from text to binary (1 for phishing, 0 for safe). Pandas can be used for column renaming, label encoding, and merging.

4.3 Cleaning the Consolidated Dataset

Since some data sets overlap (e.g., both use Enron corporation emails),the "drop_duplicates" function from Pandas can be used to remove duplicates. Emails with empty bodies or fewer than 10 characters will also be removed to reduce noise.

4.4 Data Transformation

The Hugging Face Transformers library can be imported to tokenize email bodies with a pre-trained BERT tokenizer Hugging Face (2023a). This converts text into numerical IDs, producing tensors for each email. Tokenization is required as BERT only accepts numeric input Kashyap (2024).

4.5 Training, Validation, and Testing Allocation

The scikit-learn library can be imported, with the train_test_split function used to split the dataset: 70% for training and 30% for validation/testing. Then split the 30% evenly—15% for validation and 15% for testing. This follows a proven approach for fine-tuning BERT on phishing detection Atawneh & Aljehani (2023).

5 Architecture

In the process of choosing the optimal architecture for our model, extensive research was conducted to determine which neural network is best suited for phishing detection tasks. After comparing several neural networks (refer to Appendix A for a full comparison) the primary architecture for this project will leverage a transformer-based neural network. For our project we will specifically use a pre-trained BERT modelHugging Face (2023b). The selection was made as it is well-suited for natural-language processing tasks, enhancing our models ability to understand subtle linguistic cues; which are often present in evolving phishing emails. Note that due to the demanding number of parameters required for BERT, it is prone to overfitting, especially on smaller datasets, see risk register for mitigation strategies. Our proposed system involves fine-tuning this trained BERT model which includes the following process:

1. **Input Layer:** Raw Email Text Isolation and Clean Up (removing special characters).
2. **Tokenizer Layer (via Hugging Face Tokenizer Hugging Face (2023c):** Auto-tokenizes raw text, simplifies data and fine-tunes the pre-trained BERT models.
3. **Embedding and Transformer Encoder Layer (BERT):** Converts each Token ID into dynamic vectors which then outputs contextual embedding and classifications for each token.
4. **Classification [CLS] and Vector Extraction Layer:** Only classified embedded tokens [CLS], are passed through and then this layer linearly maps these vectors to two categories "Phishing" or "Safe"
5. **Logistic Regression Layer:** Compares the results of our model against a baseline model
6. **L2 Regularization Layer:** Optimizes process by removing heavy weight data, thus preventing overfitting

***Note:** we are still in the learning stage when it comes to AI models so specific layers and parameters might be subject to change as development continues. Based on our research (Appendix A)this process seems most ideal given the functionality of our project.

6 Baseline Model

To evaluate the effectiveness of our pre-trained BERT model, we will compare its performance against a baseline logistic regression model. Logistic regression is commonly used for binary classification problems and, in this case, will be used to differentiate between phishing and "safe" emails. The process retrieved from Robu (2023) involves tokenizing the text data (text vectorization), extracting relevant features from the tokenized input (for this we can potentially use TF-IDF which

essentially adds a weight to the word based on how important the model deems it to be based on frequency), and applying a logistic function to predict the probability of each email belonging to either class. This baseline will allow us to assess whether the added complexity of BERT yields a significant performance improvement over a simpler, more interpretable model.

7 Ethical Considerations

There are several ethical considerations when developing an AI-based phishing email detection tool. The following are the considerations such as ethical considerations during data collection, limitations of the model and training data, and lastly the impact of the model.

7.1 Ethical Considerations during Data Collection

Below is a table containing the ethical considerations during data collection listed with their issue and how to mitigate the issue ethically.

Table 2: Ethical Issues and Corresponding Practices

Ethical Concern	**Issue**	**Ethical Practice**
Data Privacy & Consent	Emails contain sensitive information. Using such data without proper anonymization or concern can violate privacy rights or data protection regulations.	Make sure all data used for training is anonymized and comes from publicly available and consented sources.
Bias & Fairness	If the training set contains too many English-language phishing or domain specific language.	Create a diverse and balanced dataset. Evaluate the model using a variety of different user groups.
False Positives and Negatives	False positives can disrupt communication while false negatives pose security risks.	Tune thresholds carefully and allow human override. Provide clear error reporting to support informed decisions.
Mislabeling	Incorrect or inconsistent labelling or misclassifying legitimate emails as phishing.	Leads to model inaccuracies and may result in harmful misclassifications. To avoid this, perform regular audits and tests.
Data Security	Training data might be stored insecurely or shared inappropriately.	Breaches can expose sensitive email content and violate trust. Best practice is to implement secure storage and encryption for any dataset containing user content.

7.2 Limitations in the Model and Training Data

1. **Data Collection Bias:** The model is trained on a dataset that may not fully represent all phishing attacks, especially from non-English language domains. If the dataset is sourced from a Western source, the model would be biased towards users from different cultural and linguistic backgrounds.

2. **Privacy:** Emails contain sensitive and private information. Using it as training raises concerns about data misuse. If the model stores and uses the emails for training, it must do so under consent and data protection laws.

7.3 The impact of the Model

The implementation of this model offers notable advantages but also presents potential ethical liabilities:

- **Advantage:** It enhances user safety by identifying phishing attempts in sensitive sectors such as finance and healthcare.
- **Liability:** Without proper oversight, it could infringe on privacy by enabling the monitoring of personal communications, or be exploited by attackers to craft phishing emails that evade detection systems.

8 Project Plans

This section outlines our team's overall project plan, including how tasks will be scheduled, assigned and executed over the rest of the term. Below includes tables structured with rough estimates of internal deadlines as well for organization and workflow.

8.1 Weekly Workflow and Communication

- **Weekly meetings:** Thursdays, 3-4pm EST (Google Meet)
- **Code management:** GitHub Martins (2025) and Google Colab Chandra et al. (2025) (to avoid overwriting)
- **Task Updates:** Share progress in a shared Google doc and in iMessage group chat periodically as progress happens

8.2 Final Project Plan

The table below provides a structured project plan for the remaining of the semester.

Table 4: Final Project Plan

Task	Internal Deadline	Description
Project Proposal	June 5, 2025	Providing background information, and explaining the application of course content in the project.
Preprocessing Data (Tokenization)	June 20, 2025	Includes splitting into training/sample data.
Install and Import Hugging Face Transformers	June 20, 2025	Set up environment, test with sample data.
Baseline Model using Logistic Regression	June 27, 2025	Evaluate performance for comparison.
Apply Pretrained BERT Model	June 27, 2025	Use prebuilt models.
Fine-tune BERT (Hyperparameter Tuning)	July 11, 2025	Adjust learning rate, batch size, and other hyperparameters.
Progress Report	July 10, 2025	Provide updates on project. Explain any new findings, changes that were made, successes and failures.

Task	Internal Deadline	Description
Apply L2 Regularization	July 25, 2025	Compare performance with or without regularization.
Evaluate Model (Error, Loss, Accuracy)	August 1, 2025	Generate final metrics.
Final Configuration and Adjustments	August 8, 2025	Finalize project for submission.
Project Presentation	August 14, 2025	Create a pre-recorded video and demonstrate how the model performs.
Final Deliverable	August 14, 2025	Final report must document the entire project and match the presentation content.

8.3 Project Proposal Task Division

The table below provides a structured table regarding the work distribution for this proposal.

Table 5: Project Proposal Task Division

Task Name	Member(s)	Internal Due Date
Introduction	Asmita Chandra, Joonseo Park	June 12
Illustration	Asmita Chandra	June 12
Background and Related Work	Everyone	June 12
Data Processing	Joonseo Park	June 12
Architecture	Everyone	June 12
Baseline Model	Lee Martins	June 12
Ethical Considerations	Riya Kapoor	June 12
Project Plans	Riya Kapoor	June 12
Risk Register	Lee Martins	June 12
References	Everyone	June 12

8.4 Roles

The roles that have been distributed will remain set for the duration of the project. Should a member need help fulfilling the tasks of their role, they are **required** to reach out to another member **before** the impending deadline.

Table 7: Team Roles and Responsibilities

Role	**Description**	**Member**	**Date**
Delivery Man	Submitting deliverables on Quercus	Joonseo Park	Quercus Deadline
Meeting Coordinator	Overseeing and scheduling meetings. Responsible for cancellations and rescheduling.	Asmita Chandra	Weekly Thursday 3–4pm EST
Scribe	Keeps notes of what occurs in meetings	Lee Martins	Weekly Thursday 3–4pm EST
Editor	Oversees all final deliverables and ensures that it is aligned with rubric.	Riya Kapoor	Two days before internal deadlines

9 Risk Register

When working on any project, it is inevitable risks and issues will arise. It is imperative to employ forward thinking and mitigation strategies so this project does not feel the impact of any of the potential risks or setbacks.

9.1 Departure of a Team member

Since this course simulates industry conditions, no extensions will be given. To avoid issues, all members must share availability a week in advance and give regular updates. If someone leaves, their tasks will be fairly reassigned, with efforts made to consult them for continuity.

9.2 Model Overfitting or Poor Generalization

BERT is ideal for this project but prone to overfitting, especially on small datasets, which can hurt performance. To address this, we'll use regularization methods like L2 (weight decay), easily implemented with the Hugging Face Transformers Library Hugging Face (2023a).

9.3 Model Drift

Phishing emails are constantly evolving, especially with AI-generated content, causing patterns to shift over time and risking model drift. This could reduce model accuracy however, the risk is low due to our use of diverse datasets, including real and AI-generated emails. To further mitigate this, we recommend periodic retraining to maintain effectiveness.

9.4 Extended Training Times

BERT's high computational cost leads to longer training times, which may exceed our time limits and slow development. While accuracy won't suffer, usability may. To mitigate this, we'll train on a GPU and freeze non-essential layers, following best fine-tuning practices.

REFERENCES

AcuteShrewdSecurity. Llama-phishsense-1b [model]. https://huggingface.co/AcuteShrewdSecurity/Llama-Phishsense-1B, 2024. Accessed: 2025-06-12.

ClanX AI. Convolutional neural networks (cnns). https://clanx.ai/glossary/convolutional-neural-networks-cnns, 2023. Accessed: 2025-06-11.

Mohammad Al-Shabi, Abdul Monem Mohammed Ameen, and Mohammed M. Hamza. Email phishing detection using bert-based deep learning model. *Procedia Computer Science*, 235:1580–1587, 2024. doi: 10.1016/j.procs.2024.04.265. URL https://www.sciencedirect.com/science/article/pii/S1877050924034136.

Naser Abdullah Alam. Phishing email dataset. https://www.kaggle.com/datasets/naserabdullahalam/phishing-email-dataset, 2023. Accessed: 2025-06-11.

Jay Alammar. A visual notebook to using bert for the first time. https://colab.research.google.com/github/jalammar/jalammar.github.io/blob/master/notebooks/bert/A_Visual_Notebook_to_Using_BERT_for_the_First_Time.ipynb, 2020. Accessed: 2025-06-12.

Hattan A. Alshamrani, Ahmed H. Bayoumi, and Ahmed I. Zaghloul. Phishing email detection using hybrid features and machine learning. In *2022 International Conference on Computer and Applications (ICCA)*, pp. 41–46. IEEE, 2022. doi: 10.1109/ICCA54926.2022.9795286. URL https://ieeexplore.ieee.org/document/9795286.

Samer Atawneh and Hamzah Aljehani. Phishing email detection model using deep learning. *Electronics*, 2023. URL https://www.sciencedirect.com/science/article/pii/S1877050924034136.

Asmita Chandra, Riya Kapoor, Lee Martins, and Joonseo Park. Aps360 phishing detection project [google colab notebook]. https://colab.research.google.com/drive/1NLptCenpKZXgpdwbSJjvOvBbzKUbV3qJ?usp=sharing, 2025. Accessed: 2025-06-12.

Cyber Cop. Phishing email detection. https://www.kaggle.com/datasets/subhajournal/phishingemails, 2023. Accessed: 2025-06-12.

GeeksforGeeks. Advantages and disadvantages of ann in data mining. https://www.geeksforgeeks.org/advantages-and-disadvantages-of-ann-in-data-mining/, 2023a. Accessed: 2025-06-11.

GeeksforGeeks. Explanation of bert model (nlp). https://www.geeksforgeeks.org/explanation-of-bert-model-nlp/, 2023b. Accessed: 2025-06-11.

Francesco Greco. Human and llm-generated phishing and legitimate emails. https://www.kaggle.com/datasets/francescogreco97/human-llm-generated-phishing-legitimate-emails, 2024. Accessed: 2025-06-11.

Hugging Face. Transformers documentation. https://huggingface.co/docs/transformers, 2023a. Accessed: 2025-06-11.

Hugging Face. Bert model documentation. https://huggingface.co/docs/transformers/en/model_doc/bert, 2023b. Accessed: 2025-06-12.

Hugging Face. Tokenizer class documentation. https://huggingface.co/docs/transformers/en/main_classes/tokenizer, 2023c. Accessed: 2025-06-11.

Piyush Kashyap. Guide to tokenization and padding with bert: Transforming text into machine-readable data. https://medium.com/@piyushkashyap045/guide-to-tokenization-and-padding-with-bert-transforming-text-into-machine-readabl 2024. Accessed: 2025-06-12.

Pankaj Kumar, Akash Singh, and Ramesh Thakur. Email phishing detection using convolutional neural networks. *2024 International Conference on Computational Intelligence and Smart Communication (CISC)*, 2024. doi: 10.1109/CISC59793.2024.10872755. URL https://ieeexplore.ieee.org/document/10872755.

Lee Martins. Phishing email detection system using bert. https://github.com/leahmdmartins10/Phishing-Email-Detection-System-BERT, 2025. Accessed: 2025-06-13.

Radoslav Miltchev, Dimitar Rangelov, and Genchev Evgeni. Phishing validation emails dataset. https://research.utwente.nl/en/datasets/phishing-validation-emails-dataset, 2024. Accessed: 2025-06-11.

Milvus. What are some issues with convolutional neural networks? https://milvus.io/ai-quick-reference/what-are-some-issues-with-convolutional-neural-networks, 2023. Accessed: 2025-06-11.

Diego Miranda. A comprehensive overview of lstm networks and their limitations. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5080605, 2024. Accessed: 2025-06-11.

Neurond. What is bert? https://www.neurond.com/blog/what-is-bert, 2023. Accessed: 2025-06-11.

Rishabh Robu. Natural language processing: Linear text classification. https://medium.com/@RobuRishabh/natural-language-processing-linear-text-classification-898f64a1e04a, 2023. Accessed: 2025-06-11.

Priya Sharma. Artificial neural networks: Advantages and disadvantages. https://www.researchgate.net/publication/323665827_Artificial_Neural_Networks_Advantages_and_Disadvantages, 2018. Accessed: 2025-06-11.

Spiceworks. What is a neural network? https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-a-neural-network/#_003, 2023. Accessed: 2025-06-11.

APPENDIX

A TABLE COMPARISON

Sources used for this table: GeeksforGeeks (2023a); Sharma (2018); AI (2023); Milvus (2023); Spiceworks (2023); Miranda (2024); GeeksforGeeks (2023b); Neurond (2023)

Table 9: Comparison of Neural Network Architectures

Neural Network	**Applications**	**Pros**	**Cons**
Artificial Neural Network (ANN)	• Social media (people-you-may-know) • Artificial personal assistants • Health care systems	• Processes data in parallel • Can multitask • Operates with incomplete knowledge	• Processor-dependent parallelism • Unclear optimal structure • Requires numerical input
Convolutional Neural Network (CNN)	• Image recognition • Object detection • Facial recognition • Autonomous vehicles • Medical image analysis	• Unsupervised learning • High image accuracy • Automatic feature extraction	• Slow training times • Requires large labeled datasets • Vulnerable despite regularization • Struggles with generalization
Long Short-Term Memory (LSTM)	• Text-to-speech • Sales forecasting • Stock market prediction	• Self-learns from errors • Solves vanishing gradient • Improved memory over standard RNN	• High memory cost • Long training time • Still struggles with long-term dependencies
BERT (Transformer)	• Language generation • Text classification • Text summarization • Conversational AI	• Deep language understanding • Bidirectional context encoding • Efficient for large NLP tasks	• Risk of overfitting on small data • Requires deep NLP knowledge • Higher training time