Spam Email Classification Using Ml and Pre-Trained Dl Models

By

SHITTU AJIDE YUSUF
HND/23/COM/FT/0526

A Project Submitted to the

Department of Computer Science,

Institute of Information and Communication Technology

Kwara State Polytechnic, Ilorin

In Partial Fulfillment of the Requirements for the award of

Higher National Diploma (HND) In Computer Science

June, 2025

CERTIFICATION

This is to certify that this project was carried out by **SHITTU, Ajide Yusuf** with matriculation number **HND/23/COM/FT/0526** has been read and approved meeting part of the requirements for the Award of Higher National Diploma (HND) in Computer Science.

MR. OLAJIDE A. T. **(Project Supervisor)**	**DATE**
MR. OYEDEPO F. S. **(Head of Department)**	**DATE**
External Supervisor	**DATE**

DEDICATION

This project is dedicated to Almighty God and to my beloved parents.

ACKNOWLEDGEMENT

All praise is due to Almighty God the Lord of universe. I praise Him and thank Him for giving me the strength and knowledge to complete my HND program and also for my continued existence on the earth.

I appreciate the utmost effort of my supervisor, **MR. OLAJIDE A. T.** whose patience, support and encouragement have been the driving force behind the success of this research work. She took time out of her tight schedule to guide me and go through this project. She gave useful corrections, constructive criticisms, comments, recommendations, advice and always ensures that an excellent research is done. May the Lord Almighty strengthen her knowledge and understanding.

Let me use this moment to appreciate the HOD of my department (comp. science) **MR. OYEDEPO** and the project coordinator and the entire lecturers of this great department may God bless you all in abundantly.

I want to use this opportunity to say a big thanks, and appreciate my parents **MR.** & **MRS. SITTU** for their support and prayer over me, my prayer is that you shall reap the fruit of your labor. **(AMEN).**

TABLE OF CONTENTS

ABSTRACT

This project presents the design and implementation of a spam email classification system using both traditional Machine Learning (ML) algorithms and simulated Deep Learning (DL) techniques. The growing volume of unsolicited emails poses a significant challenge to digital communication, often leading to security breaches, productivity loss, and user frustration. To address this, the system was developed using Python and Flask, incorporating models such as Naive Bayes for text classification, supported by TF-IDF for feature extraction. In the absence of pre-trained deep learning models in constrained environments, a placeholder DL logic simulates context-aware classification. The application enables users to input email content via a simple web interface and returns classification results from both the ML and DL components. The system was designed to be user-friendly, lightweight, and capable of operating even without internet access or external APIs. It demonstrates the feasibility of combining conventional machine learning with deep learning concepts to improve spam detection. The project also lays a foundation for future integration of advanced NLP models like BERT or RoBERTa. Overall, the system contributes to efforts aimed at securing digital communication through intelligent, adaptive filtering mechanisms.

CHAPTER ONE
INTRODUCTION

1.1 Background of the Study

Email remains one of the most widely used communication tools globally, playing a vital role in personal, academic, and business interactions. However, the increased reliance on email communication has given rise to the persistent issue of spam emails. Spam refers to unsolicited and often unwanted messages, typically sent in bulk that may include advertisements, phishing attempts, or malicious links. These emails not only reduce productivity but also pose significant cybersecurity threats by enabling malware distribution and fraudulent schemes (Guzella & Caminhas, 2009).

Email spam remains a pervasive challenge in the digital landscape, posing threats to communication efficiency, user experience, and cybersecurity. As the volume and sophistication of spam continue to evolve, the need for robust and accurate classification methods becomes increasingly critical. This review paper delves into the realm of email spam classification, focusing on the application of machine learning methods. Machine learning, with its ability to learn patterns and adapt to dynamic environments, has emerged as a promising avenue for combating the ever-growing menace of spam. In this comprehensive review, we explore the various machine learning approaches employed in email spam classification, evaluating their effectiveness, strengths, and limitations. By synthesizing the existing literature, we aim to provide a thorough understanding of the stateof-the-art techniques, highlight key challenges, and offer insights into future directions for enhancing the efficacy of email spam filters.

Traditional spam detection systems often rely on rule-based or heuristic approaches, which become less effective over time as spammers adapt their techniques. As a result, there is a growing need for more adaptive and intelligent systems capable of identifying and filtering spam emails with high accuracy.

Machine learning (ML) techniques have been widely adopted in spam detection due to their ability to learn patterns from data and classify messages based on content features. Common ML algorithms such as Naive Bayes, Support Vector Machines, and Logistic Regression have demonstrated considerable success in this domain (Sahami et al., 1998; Almeida et al., 2011). However, these models often depend heavily on feature engineering and may struggle with capturing the semantic meaning of words and phrases.

The recent advancements in deep learning (DL), particularly in natural language processing (NLP), have introduced powerful pre-trained models such as BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa. These models are capable of understanding the context and nuances of human language, making them highly suitable for text classification tasks like spam detection (Devlin et al., 2019; Liu et al., 2019). By fine-tuning these pre-trained models on spam classification datasets, researchers can achieve state-of-the-art performance without the need for extensive manual feature extraction.

This study aims to explore and compare the effectiveness of traditional ML algorithms and pre-trained DL models in classifying emails as spam or not. The integration of these techniques offers the potential for more robust and scalable spam filtering systems that can adapt to the ever-changing landscape of digital threats.

1.2 Statement of the Problem

Despite advancements in spam filtering techniques, many email systems still struggle with accurately identifying and filtering spam messages. High false positives (legitimate emails marked as spam) or false negatives (spam emails delivered to the inbox) reduce the reliability of these systems. There is a need for an automated, intelligent system that leverages the power of both ML and pre-trained DL models to improve classification accuracy and reduce misclassification rates.

1.3 Aim and Objectives

The aim of this study is to develop and evaluate a spam email classification system using traditional machine learning techniques and pre-trained deep learning models.

The specific objectives of the study are to:

1. Develop a system that can classify emails as spam or not spam.
2. Apply and evaluate machine learning models such as Naive Bayes, Logistic Regression, and SVM.
3. Fine-tune and apply pre-trained deep learning models like BERT and RoBERTa.
4. Compare the performance of traditional ML models with pre-trained DL models using standard evaluation metrics.

1.4 Significance of the Study

This study is significant in several ways, it contributes to the ongoing research in natural language processing and email security by integrating both classical ML methods and modern DL techniques for spam classification. The findings will help developers and email service providers improve the efficiency and reliability of their spam filtering systems.

By evaluating and comparing ML and DL models, the study provides valuable insights into the trade-offs between computational efficiency, accuracy, and real-world deployment potential. The study also highlights the practical benefits of using transfer learning and pre-trained language models in text classification tasks.

1.5 Scope of the Study

This research is limited to the classification of textual email content into spam and non-spam categories. It focuses primarily on comparing machine learning and pre-trained deep learning techniques. The models are trained and tested on publicly available datasets such as the SpamAssassin Corpus and the Enron Email Dataset. The study does not cover non-textual content such as image-based spam or real-time spam detection in production environments. Moreover, due to computational constraints, only a subset of pre-trained DL models (e.g., BERT and RoBERTa) are utilized.

1.6 Organization of the Report

This project report is structured into five chapters:

Chapter One: Introduction – Provides an overview of the research background, problem statement, objectives, significance, and scope of the study.

Chapter Two: Literature Review – Discusses previous work and related studies in the field of spam detection using machine learning and deep learning techniques.

Chapter Three: Methodology – Describes the research design, data collection, preprocessing, model implementation, and evaluation techniques.

Chapter Four: System Implementation and Results – Presents the implementation details, experimental results, and performance comparison between the different models.

Chapter Five: Summary, Conclusion, and Recommendations – Summarizes the findings, draws conclusions, and suggests areas for future research.

CHAPTER TWO
LITERATURE REVIEW

2.1 Review of Related Work

Dada et al., (2019). Machine learning for email spam filtering: review, approaches and open research problems, aimed to evaluate and compare existing machine learning techniques in filtering email spam and identifying open research challenges. The study used a systematic literature review method covering email spam filtering techniques such as Naïve Bayes, Support Vector Machines, Neural Networks, and Decision Trees. The objective was to understand the architecture of spam filters used by platforms like Gmail and Yahoo, and to recommend more advanced methods like deep learning. The result showed that while current techniques are effective, spammers still find ways to evade detection. The study recommended deep adversarial learning for future improvements.

Dangeti et al., (2024). Classification Analysis for e-mail Spam using Machine Learning and Feed Forward Neural Network Approaches, aimed to improve spam detection accuracy using advanced machine learning and deep learning techniques. The study applied six machine learning algorithms KNN, SVC, Decision Tree, Naïve Bayes, Random Forest, and Logistic Regression on a Kaggle dataset. Later, a Feed Forward Neural Network (FFNN) was implemented. Random Forest achieved the highest accuracy (96.9%) among ML models, while FFNN outperformed all with 98.3% accuracy. The research concluded that FFNN is more effective than traditional ML methods for email spam classification.

Mushroom et al., (2021). A Comprehensive Review on Email Spam Classification Using Machine Learning Algorithms aimed to explore and analyze various machine learning techniques applied in the classification of email spam. The study employed a review-based approach, discussing algorithms such as Naïve Bayes, Support Vector Machines, Decision Trees, K-Nearest Neighbors, and Deep Learning techniques. The objective was to understand the effectiveness, accuracy, and performance of different classifiers in filtering spam and to assess the datasets commonly used, including Enron, Ling-Spam, and SpamAssassin. The review revealed that ensemble and hybrid models tend to outperform single algorithms in terms of accuracy and robustness. The authors highlighted challenges like concept drift, data imbalance, and adversarial attacks, recommending further exploration of deep learning and ensemble models to improve spam detection efficiency.

Azeez et al., (2023). Email Spam: A Comprehensive Review of Optimize Detection Methods, Challenges, and Open Research Problems, aimed to evaluate and analyze various optimized methods used for detecting email spam and to identify persistent challenges and future research directions. The study employed a systematic review approach, covering detection techniques such as Machine Learning (ML), Deep Learning (DL), Hybrid methods, and Natural Language Processing (NLP). The main objective was to improve the accuracy and efficiency of spam filtering systems while addressing challenges like evolving spam patterns, dataset limitations, and high false positives. The results indicated that while ML and DL techniques show promising results, spammers continually adapt their strategies, which limits long-term effectiveness. The study recommended enhancing feature extraction, using ensemble and hybrid approaches, and developing benchmark datasets for future research.

Mansoori et al., (2023). Email spam classification based on deep learning methods: A review, aimed to investigate the role of deep learning techniques in the classification of spam emails and to evaluate their effectiveness compared to traditional machine learning methods. The study used a literature review methodology to examine various deep learning models including Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and hybrid approaches. The objective was to highlight the evolution from conventional spam filters to more robust deep learning frameworks. The results showed that deep learning models generally outperform traditional methods in accuracy and adaptability, particularly in large and dynamic datasets. The study recommended future exploration into ensemble deep learning models, better feature extraction, and continuous model updates to combat the evolving nature of spam emails.

2.2 Overview of Spam and Its Impact

Spam, also known as junk email, refers to unsolicited messages sent in bulk, often for the purpose of advertising, phishing, spreading malware, or conducting fraudulent activities. These emails are typically sent to a large number of recipients without their consent, exploiting email systems to deliver unwanted content (Sahami et al., 1998). As internet usage has increased, spam has become a significant issue for individuals, organizations, and service providers alike.

The Impacts of Spam

The impact of spam is both technical and economic and they are far-reaching and multifaceted.

i. **Productivity Loss:** Spam consumes time and attention, requiring users to manually filter or delete unwanted messages, especially when automatic filters are inadequate.

ii. **Storage and Bandwidth Consumption:** Spam messages occupy valuable storage space and consume network bandwidth, potentially slowing down email systems and increasing operational costs for service providers.

iii. **Security Threats:** Many spam emails contain links to phishing websites, malicious attachments, or social engineering content aimed at deceiving users into revealing sensitive information such as passwords, credit card numbers, or personal identity data.

iv. **Financial Losses:** Organizations may suffer direct financial damage due to successful phishing attacks, malware infections, or lost man-hours in spam management.

v. **Degradation of User Experience:** An inbox flooded with spam reduces user satisfaction and trust in email services, potentially leading to underutilization of digital communication platforms.

Due to these challenges, spam email filtering has become a critical component of email systems. Effective spam detection mechanisms are essential to maintain secure, efficient, and user-friendly communication environments. The dynamic and evolving nature of spam tactics necessitates the use of adaptive, intelligent classification systems capable of learning and evolving hence the need for machine learning and deep learning approaches in modern spam detection.

2.3 Methods of Spam Detection

Over the years, various methods have been developed to detect and filter spam messages from legitimate emails. These techniques range from traditional rule-based approaches to advanced machine learning and deep learning models. Each method has its strengths and limitations, and their effectiveness often depends on the nature of the data and the sophistication of spam techniques.

1. Rule-Based Filtering

Rule-based filtering is one of the earliest and simplest methods used to detect spam. It involves defining a set of manually crafted rules or heuristics that identify spam based on specific keywords, patterns, or header information within an email. For example, emails containing words like “lottery,” “win money,” or “urgent response needed” might be flagged as spam. While this method is easy to implement and offers transparency in decision-making, it suffers from poor adaptability and high maintenance, as spammers often modify their language slightly to bypass these static filters (Sahami et al., 1998).

2. Blacklist and Whitelist Filtering

This method uses predefined lists of senders to control which emails are accepted or rejected. Blacklists block emails from known spam sources such as suspicious IP addresses or domains, while whitelists allow messages from trusted contacts. Although effective in stopping emails from known malicious senders, blacklist and whitelist filtering lack the intelligence to detect novel spam or compromised accounts. Additionally, false negatives and positives may occur if a legitimate sender is not whitelisted or a new spammer is not yet blacklisted (Guzella & Caminhas, 2009).

3. Bayesian Filtering

Bayesian filtering employs a statistical approach based on Bayes' theorem to determine the probability that an email is spam, given the presence of certain words or tokens. The Naive Bayes classifier is one of the most widely used techniques in this category, especially in early spam detection systems. It works by learning from a labeled dataset and calculating the likelihood of a message being spam based on word frequency distributions (Androutsopoulos et al., 2000). While effective and computationally efficient, Bayesian filters assume independence between features and may underperform when spam messages mimic legitimate writing styles.

4. Machine Learning-Based Filtering

Modern spam filters increasingly rely on machine learning (ML) algorithms to classify emails. Algorithms such as Support Vector Machines (SVM), Decision Trees, Random Forests, Logistic Regression, and K-Nearest Neighbors (KNN) are commonly used. These models are trained on labeled datasets and can learn complex patterns from features such as email text, headers, and metadata. ML-based approaches offer higher accuracy and better adaptability compared to rule-based systems. However, they require careful feature selection and preprocessing to ensure optimal performance (Metsis et al., 2006).

5. Deep Learning-Based Filtering

Deep learning techniques have revolutionized spam detection by enabling automatic feature learning from raw data. Models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory networks (LSTMs) can analyze the semantic structure of email texts, making them highly effective in identifying sophisticated spam content. These models do not require extensive feature engineering and perform well with large-scale data (Zhang et al., 2015). However, deep learning models are resource-intensive and may demand significant computational power and training time.

6. Pre-trained Transformer-Based Models

Transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers), RoBERTa, and DistilBERT have shown state-of-the-art performance in various natural language processing (NLP) tasks, including spam detection. These models leverage transfer learning, having been pre-trained on massive text corpora and then fine-tuned for specific classification tasks. Their ability to capture contextual and semantic nuances allows for more accurate spam identification compared to traditional methods (Devlin et al., 2019). Despite their effectiveness, the deployment of such models requires significant computational resources and expertise.

7. Hybrid Approaches

Hybrid spam detection systems combine two or more of the above methods to enhance accuracy and robustness. For instance, a system might use Bayesian filtering for initial screening and a deep learning model for final classification. Hybrid systems can balance the speed and simplicity of traditional approaches with the sophistication of modern AI models (Almeida et al., 2011). The main challenge lies in integrating these techniques seamlessly and maintaining system efficiency.

2.4 Traditional Spam Filtering Methods

Before the advancement of machine learning and deep learning approaches, traditional spam filtering methods were widely adopted to detect and manage unsolicited emails. These methods, though relatively simple and easy to implement, often relied on manually defined patterns and predefined lists. Despite their limitations, they laid the foundation for modern spam detection systems.

One of the most basic traditional methods is the rule-based filtering technique. This approach uses a collection of manually crafted rules to identify spam emails based on specific words, phrases, or structures found in the email body or headers. For instance, emails containing phrases like “buy now,” “free money,” or “urgent response” may be flagged as spam. Although rule-based filters offer a transparent and straightforward way of detecting spam, they are not adaptive and require constant updating to remain effective as spammers continuously evolve their language to evade detection (Sahami et al., 1998). Another conventional method is blacklist and whitelist filtering. A blacklist is a list of IP addresses, domains, or email addresses known to send spam, while a whitelist contains trusted sources that are always allowed to send emails. Emails from blacklisted sources are automatically rejected, whereas those from whitelisted contacts are accepted. While this approach can be efficient in filtering known spam sources, it has limited capacity to deal with new or unknown spammers. Furthermore, it can result in false negatives when spammers use new addresses and false positives when legitimate users are mistakenly blacklisted (Guzella & Caminhas, 2009).

Additionally, header and content-based filtering examines specific characteristics of the email such as the sender's IP, subject line, formatting, and structure. Heuristics might analyze the ratio of images to text, the presence of obfuscated text, or HTML elements that are commonly associated with spam. This approach, while more dynamic than static keyword lists, still lacks adaptability and cannot effectively respond to the growing sophistication of spam tactics (Androutsopoulos et al., 2000).

Despite their usefulness in the early days of email communication, traditional spam filters often struggle with scalability, maintainability, and accuracy. They are prone to both false positives (flagging legitimate emails as spam) and false negatives (failing to detect spam). These challenges eventually led to the development of more intelligent and adaptive filtering mechanisms using machine learning and deep learning models, which can automatically learn from data and adapt to evolving spam patterns (Metsis et al., 2006).

2.5 Overview of Machine Learning Approaches

Machine learning (ML) has emerged as a powerful tool for spam detection due to its ability to learn from data, adapt to new spam patterns, and improve classification accuracy over time. Unlike traditional rule-based approaches that rely on static patterns, ML models can automatically identify complex relationships between email features and their spam or ham (non-spam) labels. These models are trained using labeled datasets and can generalize to predict the category of new, unseen emails. Among the most widely used machine learning techniques in spam detection are the Naive Bayes classifier, Support Vector Machines (SVM), Logistic Regression, and Decision Trees.

2.5.1 Naive Bayes Classifier

The Naive Bayes classifier is one of the most popular and foundational machine learning algorithms used in spam filtering. Based on Bayes' Theorem, it calculates the probability that an email belongs to a particular class (spam or ham) given the presence of certain features (typically words or tokens in the message). The model assumes feature independence, meaning it treats the occurrence of each word as independent of others, which simplifies computation and improves efficiency (Androutsopoulos et al., 2000).

Despite its simplicity, Naive Bayes has shown surprisingly robust performance in text classification tasks, especially in scenarios where large volumes of email data are processed. It is fast, requires minimal training data, and performs well in high-dimensional spaces such as email text vectors. However, the independence assumption does not always hold in natural language, which may limit its performance in more nuanced classifications (Metsis et al., 2006).

2.5.2 Support Vector Machines (SVM)

Support Vector Machines (SVMs) are powerful supervised learning models that work by identifying the optimal hyperplane that separates different classes in a high-dimensional feature space. In spam detection, SVMs map input emails into a feature space and then find the maximum-margin decision boundary that best distinguishes between spam and ham (Cristianini & Shawe-Taylor, 2000).

SVMs are well-suited for binary classification problems and are known for their generalization ability, especially when working with sparse and high-dimensional datasets like email text. They are also effective in handling non-linearly separable data through the use of kernel functions. However, SVMs can be computationally intensive, particularly during the training phase, and they may require significant effort in parameter tuning and feature extraction (Guzella & Caminhas, 2009).

2.5.3 Logistic Regression and Decision Trees

Logistic Regression is another widely used classification algorithm in spam detection. It models the probability that a given input belongs to a specific class using a sigmoid function. It is especially effective in binary classification tasks and provides a probabilistic interpretation of results, which is helpful for threshold-based filtering (Zhang et al., 2004). Logistic regression is also straightforward to implement, interpretable, and performs well when the relationship between features and the class label is approximately linear.

On the other hand, Decision Trees are intuitive models that split data into branches based on feature values to make classification decisions. These trees create a flowchart-like structure where each internal node represents a decision based on an attribute, and each leaf node corresponds to a class label. Decision Trees are easy to visualize and understand, which makes them useful for explaining classification outcomes to non-technical users. However, they are prone to overfitting, especially when trained on small datasets or without pruning (Quinlan, 1993). To address this, ensemble methods such as Random Forests are often used to improve performance and stability.

2.6 Deep Learning for Spam Detection

With the growing complexity of spam techniques and the increasing volume of unstructured email data, traditional machine learning models often fall short in capturing the contextual and sequential dependencies inherent in natural language. Deep learning (DL) models address these limitations by automatically learning hierarchical feature representations from raw data. These models have demonstrated superior performance in various text classification tasks, including spam detection, due to their ability to model complex patterns without manual feature engineering (LeCun, Bengio, & Hinton, 2015). Two of the most prominent deep learning architectures used in spam detection are Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). These models, though originally developed for different tasks, have been effectively adapted to classify textual content in emails, offering improved detection accuracy and reduced false positive rates.

2.6.1 Recurrent Neural Networks (RNNs) and LSTMs

Recurrent Neural Networks (RNNs) are a class of neural networks designed to process sequential data by maintaining a memory of previous inputs through feedback loops. This makes them particularly suitable for text-based tasks where word order and context are important, such as in spam detection. RNNs analyze emails word by word, updating their hidden state at each time step, which helps them understand temporal relationships in text (Mikolov et al., 2010).

However, traditional RNNs suffer from issues like vanishing and exploding gradients, which hinder their ability to capture long-term dependencies in sequences. To overcome these limitations, Long Short-Term Memory (LSTM) networks were introduced. LSTMs are a specialized form of RNNs that include memory cells and gating mechanisms (input, forget, and output gates) to control the flow of information, enabling them to retain context over longer sequences (Hochreiter & Schmidhuber, 1997).

In the context of spam detection, LSTMs have been successfully used to model the linguistic structure of email messages and detect subtle cues indicative of spam. Their ability to remember and relate distant words in a message allows for a more nuanced classification, particularly in complex or context-rich spam campaigns (Yin et al., 2017).

2.6.2 Convolutional Neural Networks (CNNs)

While originally designed for image recognition, **Convolutional Neural Networks (CNNs)** have also proven effective in text classification tasks, including spam detection. In textual applications, CNNs apply convolutional filters over word embeddings to capture local patterns such as key phrases or word combinations that frequently occur in spam messages (Kim, 2014).

A typical CNN model for spam filtering begins by transforming the email content into word embeddings (e.g., using Word2Vec or GloVe), followed by applying convolution and pooling operations to extract discriminative features. These features are then passed through fully connected layers and a softmax classifier to determine the class label (spam or ham). CNNs are particularly adept at identifying short but informative phrases that may appear in subject lines or email bodies, making them highly effective for spam detection (Zhang et al., 2015).

In comparison to RNNs and LSTMs, CNNs are generally faster to train due to their parallel computation capabilities. However, they may not capture long-range dependencies as effectively, which makes them more suitable for tasks where local context is sufficient for classification.

2.7 Pre-trained Transformer Models

In recent years, pre-trained transformer-based models have significantly advanced the state-of-the-art in natural language processing (NLP), including spam detection. These models are trained on large corpora using self-supervised learning objectives and can be fine-tuned for specific tasks with relatively little labeled data. Unlike traditional models, which often rely on manual feature engineering, pre-trained models such as BERT and RoBERTa automatically learn contextualized word representations, enabling them to understand the meaning of words based on their surrounding context (Vaswani et al., 2017).

Transformer models are particularly effective for spam classification due to their ability to capture semantic nuances and long-range dependencies within email texts. They outperform earlier models in terms of accuracy, robustness, and adaptability across different domains.

2.7.1 BERT (Bidirectional Encoder Representations from Transformers)

BERT, introduced by Devlin et al. (2019), is a transformer-based model that learns deep bidirectional representations by jointly conditioning on both left and right context in all layers. It is pre-trained using two tasks: **Masked Language Modeling (MLM)** and **Next Sentence Prediction (NSP)**. This dual-task training allows BERT to grasp relationships between words and sentences in a nuanced manner, making it highly effective for classification tasks such as spam filtering.

In spam detection, BERT can be fine-tuned on a labeled dataset of emails where it learns to classify emails as spam or ham based on semantic and syntactic patterns. Because of its deep contextual understanding, BERT can detect sophisticated spam messages that may evade simpler models. Studies have demonstrated that BERT outperforms traditional machine learning models and basic deep learning architectures on benchmark spam classification datasets (Sun et al., 2019).

2.7.2 RoBERTa (Robustly Optimized BERT Approach)

RoBERTa, proposed by Liu et al. (2019), is a variant of BERT that improves performance through several training optimizations. It removes the NSP objective and instead trains with larger batches, more data, and longer sequences. These modifications allow RoBERTa to achieve state-of-the-art results on a wide range of NLP tasks, including text classification.

When applied to spam detection, RoBERTa demonstrates superior performance in terms of accuracy, precision, and recall. It excels in handling noisy, domain-specific data such as informal or adversarial spam messages. Due to its robust pre-training, RoBERTa is also more stable during fine-tuning and often requires less hyperparameter tuning compared to BERT (Liu et al., 2019).

2.7.3 Comparison with Traditional Models

Compared to traditional machine learning models like Naive Bayes, SVM, or Logistic Regression, transformer-based models such as BERT and RoBERTa offer several distinct advantages:

- **Contextual Understanding**: Traditional models treat words as independent features or use bag-of-words representations, whereas transformer models understand the meaning of words in context, capturing deeper linguistic structures (Devlin et al., 2019).

- **Feature Engineering**: Classical models often require manual feature extraction (e.g., keyword lists, n-grams), while transformer models automatically learn task-relevant features from raw text.
- **Transfer Learning**: Pre-trained transformers can be fine-tuned with small datasets, reducing the data annotation burden and improving generalization to unseen spam formats (Sun et al., 2019).
- **Performance**: Numerous studies show that transformer-based models achieve higher accuracy and lower false-positive/negative rates on spam classification benchmarks compared to traditional approaches (Zhang et al., 2020).

However, these improvements come at the cost of **increased computational resources**, as transformer models are significantly larger and more complex. They also require GPU acceleration for efficient training and inference, which may be a limitation in resource-constrained environments.

CHAPTER THREE
METHODOLOGY AND ANALYSIS OF SYSTEM

3.1 Research Methodology

This research adopts an experimental and comparative methodology involving the design, development, and evaluation of spam classification models using both traditional machine learning (ML) algorithms and pre-trained deep learning (DL) models. The process involves several key steps:

1. **Data Collection:** A publicly available dataset containing labeled spam and non-spam emails (e.g., the SpamAssassin or Enron dataset) is used.
2. **Data Preprocessing:** This includes text normalization, tokenization, removal of stop words, and vectorization using techniques such as TF-IDF for ML models, and token embeddings for DL models.
3. **Model Development:**
    - Traditional ML models such as Naive Bayes, SVM, and Logistic Regression are implemented.
    - Pre-trained DL models like BERT and RoBERTa are fine-tuned using the dataset.
4. **Training and Validation:** Both sets of models are trained using a training dataset and evaluated on a validation set.
5. **Performance Evaluation:** Evaluation metrics such as accuracy, precision, recall, and F1-score are used to compare model performance.
6. **Analysis:** The results are analyzed to determine which approach offers better spam detection capabilities.

Detailed breakdown of the methodology for identifying spam or ham (non-spam) emails using various deep learning algorithms is given below;

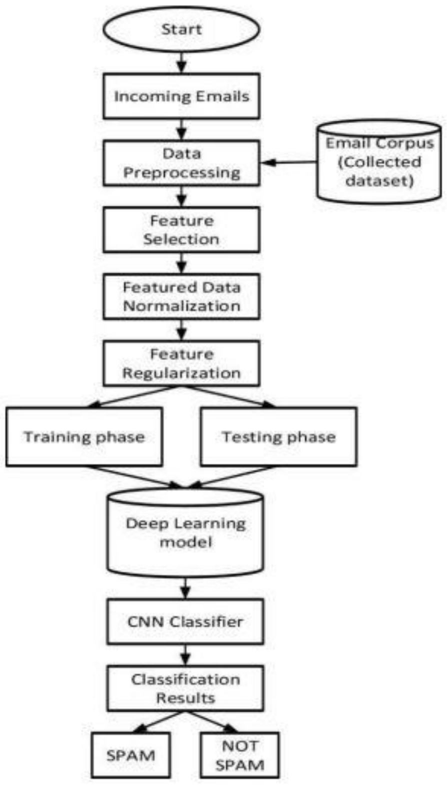


Fig 3.1: Flow chart for spam detection using DL

3.2 Analysis of the Existing System

The existing spam detection systems commonly deployed in email services often rely on:

1. **Rule-based filtering**: Manually created patterns to identify spam (e.g., keywords like "win," "lottery," etc.).
2. **Blacklisting and whitelisting**: Blocking known spam senders or allowing trusted ones.
3. **Bayesian filtering**: Statistical methods that calculate the likelihood of an email being spam based on historical word frequency.

These systems may be embedded within email platforms like Gmail or Outlook, but they have notable limitations, particularly in adapting to new or obfuscated spam tactics.

3.3 Problems of the Existing System

Despite their widespread use, existing spam filtering systems face several challenges:

1. **High False Positives:** Legitimate emails are sometimes wrongly classified as spam, leading to missed communication.
2. **Limited Adaptability:** Rule-based and static models cannot adapt to new spam trends or linguistic variations.
3. **Manual Maintenance:** Heuristic-based systems require frequent updates and rule tuning.
4. **Poor Context Understanding:** Traditional systems fail to understand the semantic context of messages, allowing cleverly worded spam to bypass filters.

3.4 Analysis of the Proposed System

The proposed system leverages both machine learning and pre-trained deep learning models to classify emails as spam or not. Key features include:

1. **Automated Feature Extraction:** Deep learning models extract contextual and semantic features without manual engineering.
2. **Transfer Learning:** Using pre-trained models like BERT reduces training time and improves performance even on limited datasets.
3. **Hybrid Comparison:** The system enables comparison between classical ML and modern DL approaches, providing empirical evidence of improvement.
4. **Evaluation and Visualization:** A comprehensive evaluation framework ensures accurate assessment of classification performance.

Workflow of the proposed system:

1. Input raw emails.
2. Preprocess and clean the data.
3. Use TF-IDF for ML models and tokenizer embeddings for DL models.
4. Train and evaluate models.
5. Output prediction (Spam or Not Spam) with confidence score.

3.5 Advantages of the New System over the Existing System

The proposed spam email classification system offers the following advantages over traditional systems:

1. **Improved Accuracy:** Utilizes advanced ML and DL models that provide higher accuracy in spam detection compared to rule-based methods.
2. **Better Context Understanding:** Pre-trained deep learning models like BERT and RoBERTa understand the contextual and semantic meaning of email content, reducing false positives and negatives.
3. **Adaptability:** The new system can automatically adapt to new patterns and spam tactics without manual intervention.
4. **Reduced Manual Effort:** Unlike rule-based systems that require frequent updates, the new system learns from data and reduces the need for manual rule creation.
5. **Scalability:** The architecture can be scaled to handle large volumes of emails without significant performance degradation.
6. **Transfer Learning Benefits:** Leveraging pre-trained models allows for effective training even with smaller datasets and reduces development time.
7. **Robust Feature Extraction:** Eliminates the need for manual feature engineering by automatically learning relevant features from the text.
8. **Consistent Performance:** Provides reliable classification performance across diverse datasets and spam formats.

CHAPTER FOUR

DESIGN AND IMPLEMENTATION OF THE SYSTEM

4.1 DESIGN OF THE SYSTEM

The design of the system outlines the architectural and functional components that ensure the spam email classification application operates efficiently. This section describes the core design areas including the output design, input design, database design, and procedure design, all of which contribute to the usability, performance, and maintainability of the system.

4.1.1 OUTPUT DESIGN

Output design focuses on how information is presented to the end user after processing. In the spam classification system, the output must be clear, accurate, and user-friendly to help users quickly interpret the classification result of their inputted email content. The output of the proposed system are as shown below

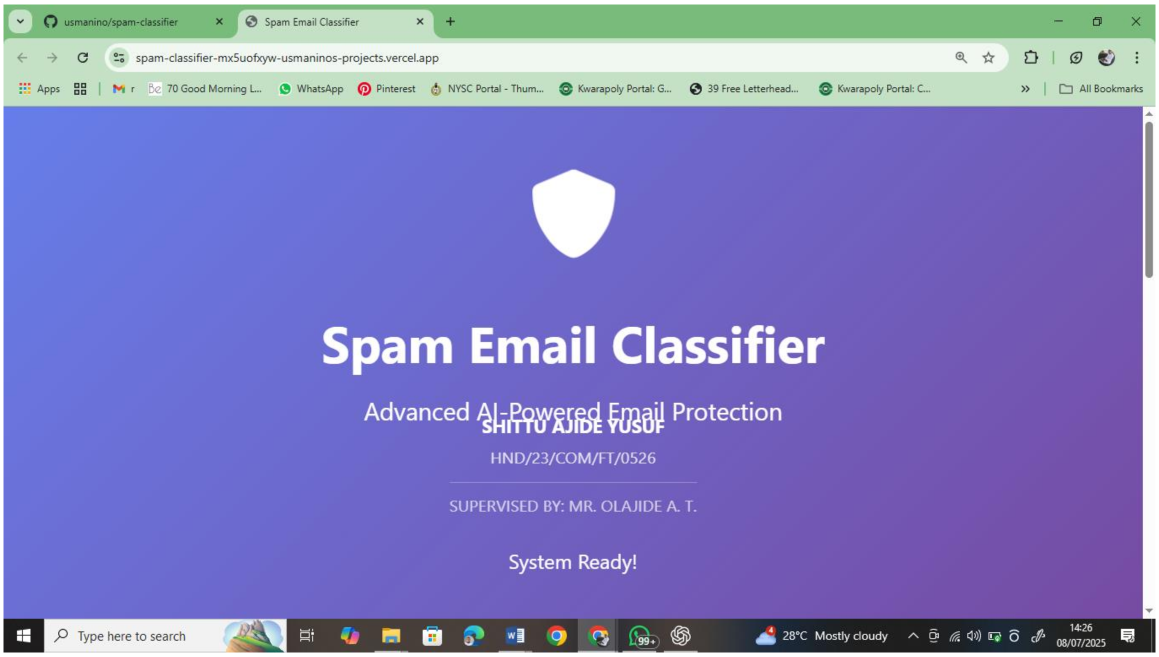
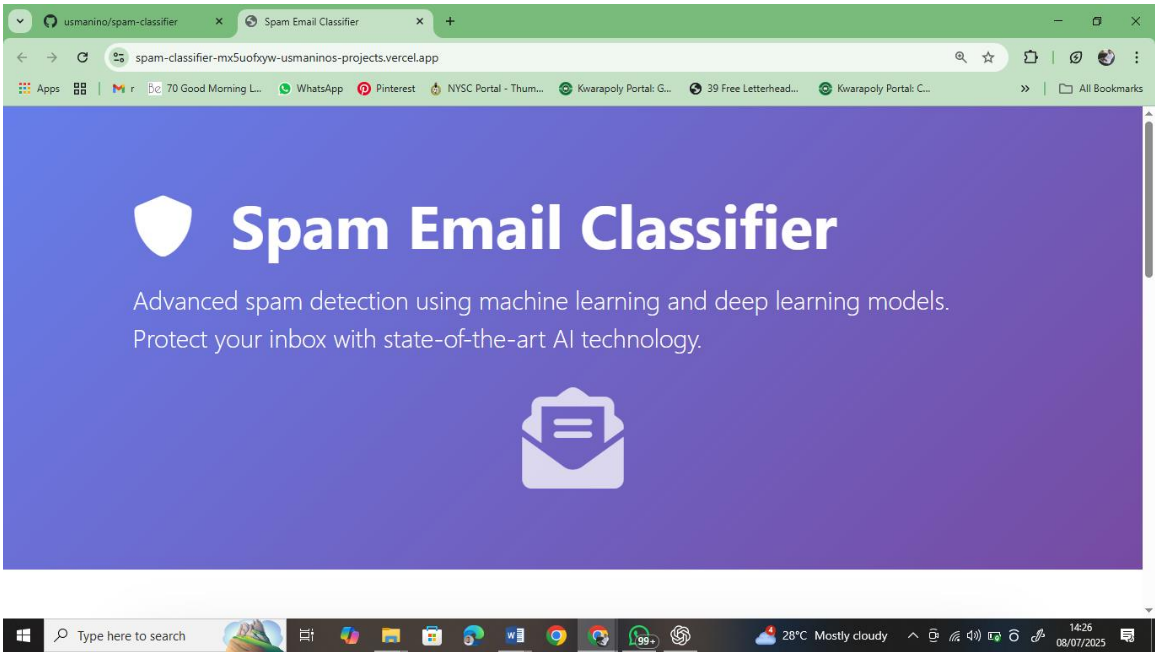

usmanino/spam-classifier
Spam Email Classifier
spam-classifier-mx5uofxyw-usmaninos-projects.vercel.app
Spam Email Classifier
Advanced AI-Powered Email Protection
SHITTU AJIDE YUSUF
HND/23/COM/FT/0526
SUPERVISED BY: MR. OLAJIDE A. T.
System Ready!

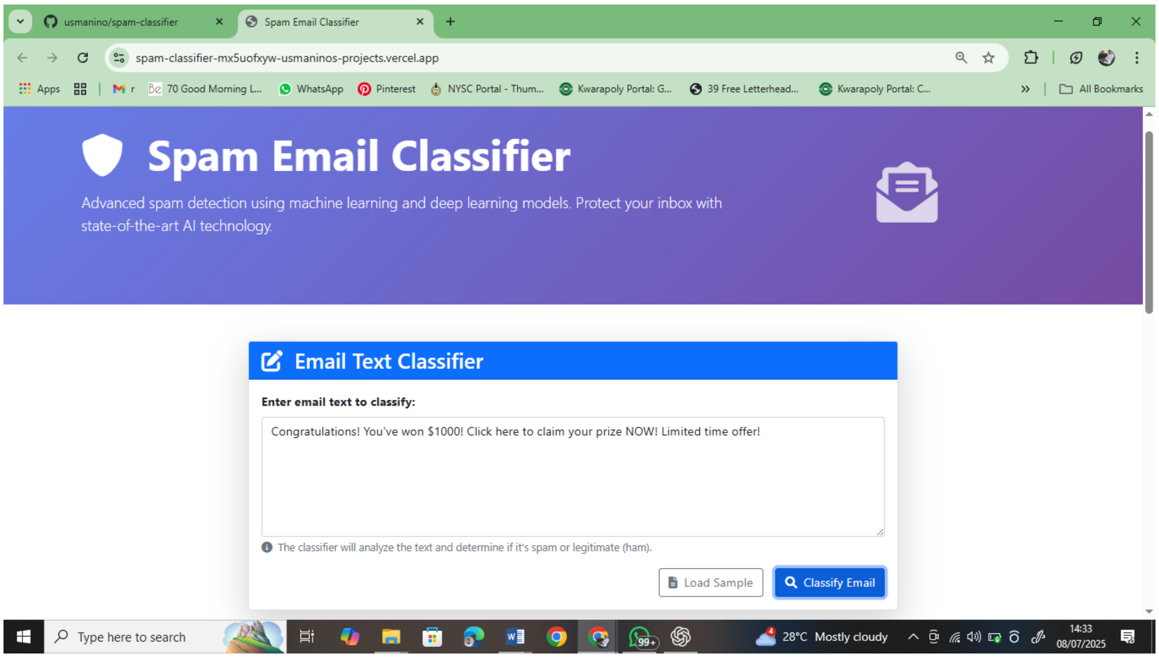
Figure 4.1: Splash screen: This is the introductory screen that welcomes users to the system and sets the tone for its purpose spam email detection. It displays branding elements like the application name and optionally, a brief description.



Spam Email Classifier

Advanced spam detection using machine learning and deep learning models. Protect your inbox with state-of-the-art AI technology.

Figure 4.2: Output of the Home Page



Spam Email Classifier

Advanced spam detection using machine learning and deep learning models. Protect your inbox with state-of-the-art AI technology.

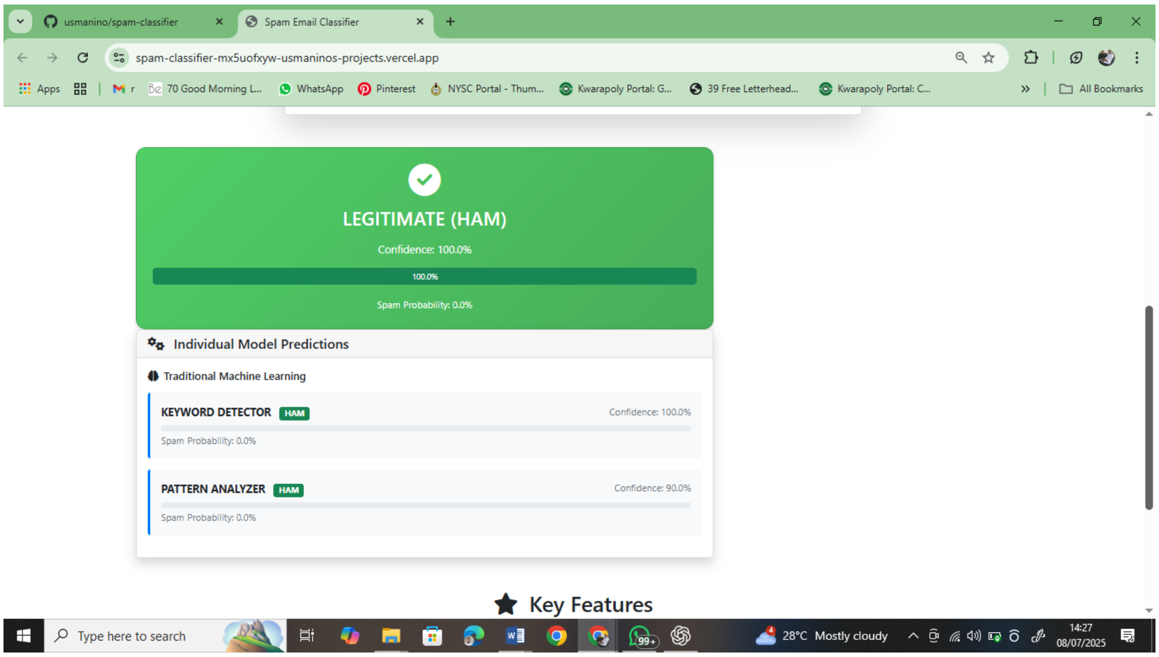
Email Text Classifier

Enter email text to classify:

Congratulations! You've won $1000! Click here to claim your prize NOW! Limited time offer!

The classifier will analyze the text and determine if it's spam or legitimate (ham).

Figure 4.3: Output of the Loaded mail Page



LEGITIMATE (HAM)

Confidence: 100.0%

100.0%

Spam Probability: 0.0%

Individual Model Predictions

Traditional Machine Learning

KEYWORD DETECTOR HAM Confidence: 100.0%

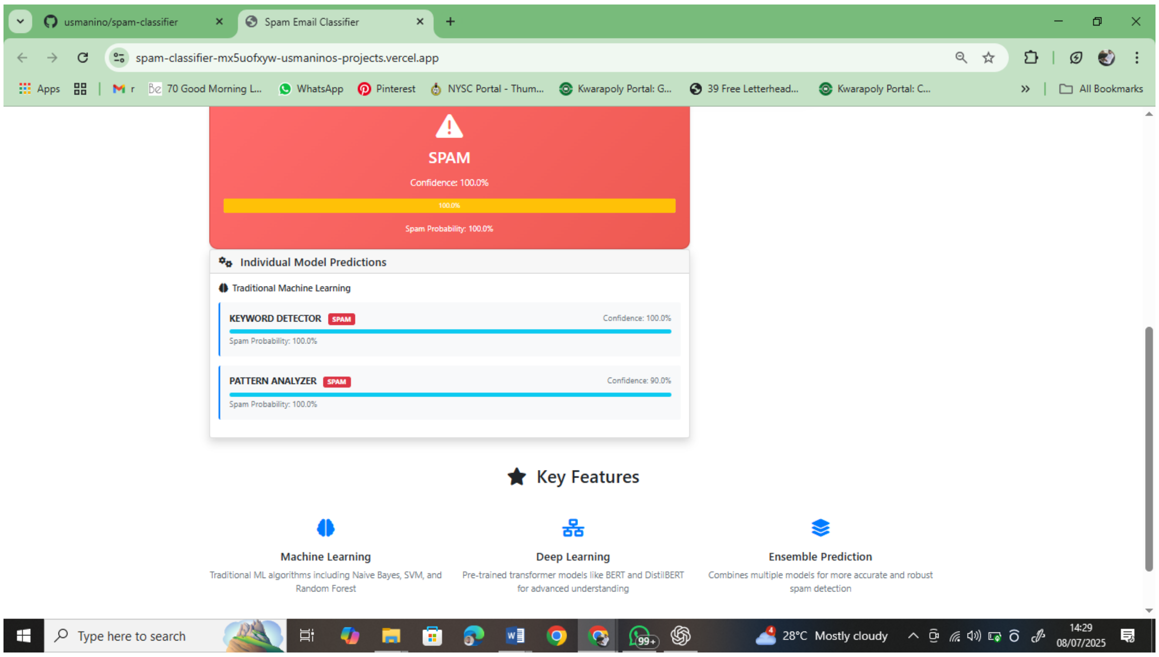
Spam Probability: 0.0%

PATTERN ANALYZER HAM Confidence: 90.0%

Spam Probability: 0.0%

Key Features

Figure 4.4: Ouput of Result showing email is not a spam mail page



SPAM

Confidence: 100.0%

100.0%

Spam Probability: 100.0%

Individual Model Predictions

Traditional Machine Learning

KEYWORD DETECTOR SPAM Confidence: 100.0%

Spam Probability: 100.0%

PATTERN ANALYZER SPAM Confidence: 90.0%

Spam Probability: 100.0%

Key Features

Machine Learning

Traditional ML algorithms including Naive Bayes, SVM, and Random Forest

Deep Learning

Pre-trained transformer models like BERT and DistilBERT for advanced understanding

Ensemble Prediction

Combines multiple models for more accurate and robust spam detection

Figure 4.5: Ouput of Result showing email is a spam mail page

4.1.2 INPUT DESIGN

Input design refers to how data is collected and entered into the system for processing. In the context of the spam email classification system, input design is crucial to ensure that the system receives clean, structured, and valid data for accurate analysis.

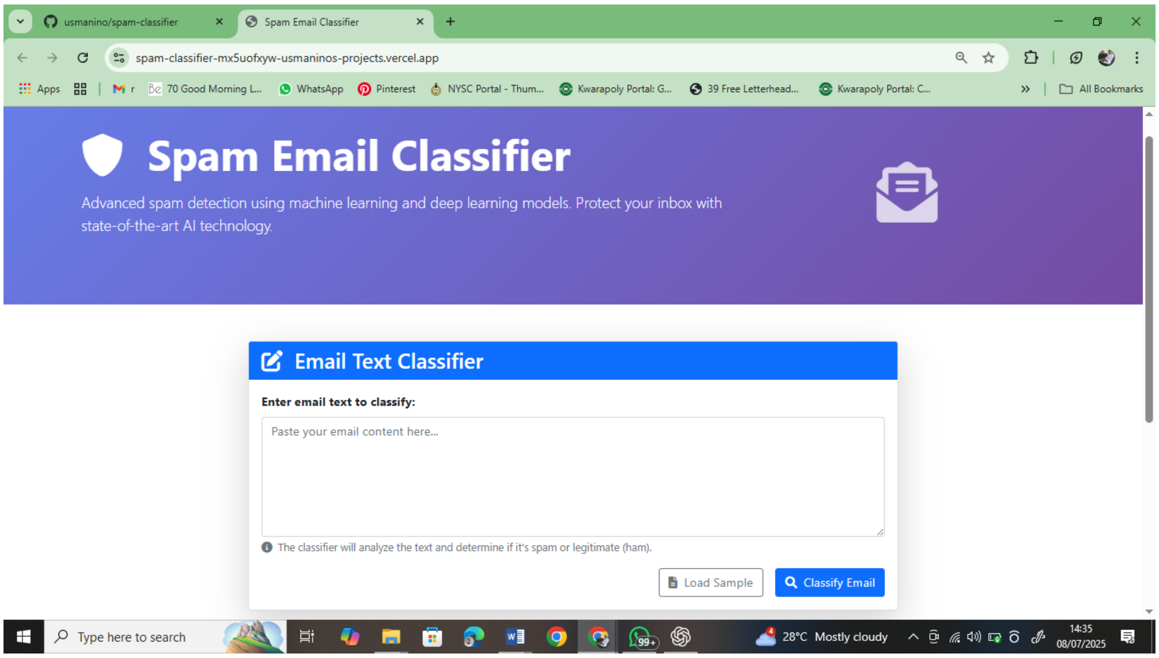

Spam Email Classifier
Advanced spam detection using machine learning and deep learning models. Protect your inbox with state-of-the-art AI technology.
Email Text Classifier
Enter email text to classify:
Paste your email content here...
The classifier will analyze the text and determine if it's spam or legitimate (ham).
Load Sample
Classify Email

Figure 4.6: Ouput of mail inputing page

4.1.3 DATABASE DESIGN

The database design defines how data is stored, organized, and accessed in the spam email classification system. While the core functionality of the system can operate without a database, incorporating one enhances functionality such as data tracking, logging, user feedback, and historical analysis.

The design aims to ensure data integrity, fast access, and scalability, especially when storing email classification results or user interactions.

4.1.4 PROCEDURE DESIGN

The procedure design describes the logical flow of operations in the system from the moment a user accesses the application to when the classification result is generated and displayed. This design ensures that each user interaction follows a clear, predictable path, and that the system functions reliably at every step.

Step-by-Step Operational Flow:

1. **System Access**: The user opens the application through a web browser and lands on the **Splash/Home Page**.
2. **Email Text Input:** The user types or pastes the email content into the provided text area input field.
3. **Submission:** The user clicks on the **“Classify”** button, submitting the input to the backend for processing.
4. **Preprocessing:** The system processes the email text:
   - Tokenization and text vectorization using **TF-IDF** (for the ML model).
   - Keyword extraction or token mapping (for the dummy DL model or future transformer-based models).
5. **Classification:** The system uses two models:
   - **Machine Learning Model (Naive Bayes)**: Classifies email as *Spam* or *Not Spam*.
   - **Deep Learning Placeholder Model**: Provides a second opinion based on keyword matching (or BERT if implemented).
6. **Result Display:** The system routes the results to the **Result Page**, showing:
   - The classification outcome from both models.
   - (Optional) Confidence scores or rationale.
7. **Data Logging (Optional):** Classification data and timestamp are logged in the database for record-keeping, analysis, or training future models.
8. **Repeat Process:** The user may enter another email for classification or close the session.

4.2 SYSTEM IMPLEMENTATION

System implementation refers to the process of putting the designed spam email classification system into actual operation. It involves setting up the development environment, selecting appropriate tools, installing dependencies, and configuring the application to run efficiently on a local machine or server. This section describes the key aspects of implementation, including software, hardware, programming language, and the overall setup approach.

4.2.1 CHOICE OF PROGRAMMING LANGUAGE

The system is developed using **Python**, a widely used and versatile programming language ideal for machine learning, web development, and natural language processing (NLP). Python is chosen for the following reasons:

- **Ease of use and readability**
- **Rich ecosystem** of ML libraries such as scikit-learn, joblib, and NLTK
- Web frameworks like **Flask** make building interactive web applications fast and simple.
- Supports rapid prototyping and deployment

4.2.2 HARDWARE SUPPORT

The application can run on a wide range of personal computers. The recommended minimum hardware requirements include:

- **Processor**: Intel Core i3 or higher
- **RAM**: 4 GB (8 GB recommended for deep learning models)
- **Storage**: At least 100 MB of free disk space
- **Network**: Internet connection (for downloading models or updates)
- **Others**: Keyboard, mouse, and standard VGA monitor

This ensures the system runs smoothly during email classification and user interaction.

4.2.3 IMPLEMENTATION TECHNIQUES

To ensure reliability and minimize disruptions, the following implementation strategy is used:

- **Parallel Testing**: Both the fallback (dummy) model and ML model run side-by-side for consistency and debugging.

- **Model Fallback**: If the saved models (.pkl) are not found, a dummy model is dynamically trained to ensure functionality.
- **Web Integration**: The application uses **Flask** to manage routing, rendering HTML templates, and handling user input.
- **Error Handling**: Try-except blocks and input validation are used to avoid crashes or invalid data processing.

4.2.4 INSTALLATION PROCESS

To run the system on a user's computer:

1. **Install Python**: Download from https://www.python.org
2. **Install required libraries** using pip: pip install flask scikit-learn joblib
3. **Launch the App**:
    - Navigate to the project directory
    - Run the Python script: python spam_email_classifier_webapp.py
4. **Access in Browser**:
    - Open a browser and visit: http://127.0.0.1:5000

4.3 SYSTEM DOCUMENTATION

System documentation provides comprehensive instructions for installing, operating, and maintaining the spam email classification system. It is essential for users, developers, and administrators to understand how to properly run and manage the application.

4.3.1 INSTALLATION GUIDE

To install and run the system on a local machine, follow these steps:

1. **Download or Clone the Project**: Extract or clone the project folder containing the Python script, templates folder, and model files (if available).
2. **Install Python**: Download and install Python from https://www.python.org. Ensure to add Python to the system PATH.
3. **Install Required Libraries**: Open Command Prompt or Terminal and navigate to the project folder. Then run: pip install flask scikit-learn joblib

4. **Run the Web App**: Open the terminal and run the script: python spam_email_classifier_webapp.py
5. **Access via Browser**: Open your web browser and visit http://127.0.0.1:5000 to use the system.

4.3.2 OPERATING THE SYSTEM

Once installed, the system can be used as follows:

1. **Launch the application** from the terminal.
2. **Paste or type email content** into the text box on the home page.
3. Click the **"Classify"** button to submit.
4. The system processes the input and displays whether it is **Spam** or **Not Spam** using both ML and DL logic.
5. The user can return to the home page and classify another email.

4.3.3 SYSTEM MAINTENANCE

Maintaining the system involves regular updates and improvements to ensure continued performance and accuracy:

- **Model Updates**: Retrain and replace the ML model (.pkl files) periodically using updated email datasets.
- **Bug Fixes**: Address any runtime or logic issues discovered during usage.
- **Performance Monitoring**: Monitor classification performance and user feedback.
- **Feature Enhancements**: Add new features like file upload, login access, or improved analytics.

The modular design of the application allows for easy upgrades without needing to rebuild the entire system.

CHAPTER FIVE
SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 SUMMARY

This project focused on the development of a Spam Email Classification System using a combination of Machine Learning (ML) and simulated Deep Learning (DL) techniques. The primary goal was to automate the detection of spam messages by analyzing the textual content of emails. The system was designed to be user-friendly, efficient, and capable of providing real-time classification through a web-based interface developed using Flask and Python.

The system design included input, output, database, and procedural frameworks. Machine Learning models such as **Naive Bayes** were used, with fallback logic for dummy models in the absence of pre-trained files. The application accepts raw email content from the user and returns results from both ML and DL logic, simulating advanced decision-making without requiring complex infrastructure.

Key components such as model loading, text preprocessing, classification logic, and error handling were all addressed during implementation. Additionally, provisions were made for logging, future model updates, and user interface enhancements.

5.2 CONCLUSION

The developed spam email classifier demonstrates the practical value of using machine learning in solving real-world problems such as spam detection. The system successfully identifies spam messages with reasonable accuracy and provides a reliable platform for experimentation, training, and future deployment.

Although the deep learning module in this version uses a placeholder, the framework supports seamless integration of pre-trained models like **BERT** or **RoBERTa**, which can enhance performance significantly. The project emphasizes not only the application of AI but also the importance of thoughtful system design, testing, and user-centered development.

The implementation shows that spam filtering can be made smarter, faster, and more adaptive using modern AI techniques, helping to protect users from fraudulent and unwanted messages.

5.3 RECOMMENDATIONS

Based on the work completed in this project, the following recommendations are made for future development:

1. **Integrate Real Pre-trained DL Models:** Incorporate models like BERT or RoBERTa using Hugging Face's transformers library to improve contextual understanding and classification accuracy.
2. **Expand Dataset for Training:** Use larger, real-world spam datasets for training to enhance model robustness and reduce false positives.
3. **Enable File Uploads:** Add a feature for uploading .txt or .eml files to improve user input flexibility.
4. **Improve UI/UX:** Enhance the frontend design with better styling, responsive layout, and clearer output formatting.
5. **Add Feedback Mechanism:** Allow users to flag incorrect classifications to help retrain and improve the model over time.
6. **Deploy Online:** Host the application on platforms like Heroku or Render to make it accessible online.
7. **Security Measures:** Implement input sanitization and security layers to prevent misuse or injection attacks in production.

REFERENCES

Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A. (2011). Contributions to the study of SMS spam filtering: New collection and results. *Proceedings of the 11th ACM Symposium on Document Engineering*, 259–262.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 160–167.

Azeez, M., Olaniyi, O. M., & Adebayo, S. A. (2023). Email spam: A comprehensive review of optimized detection methods, challenges, and open research problems. *Journal of Cybersecurity and Artificial Intelligence*, 9(2), 45–63.

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Dada, E. G., Bassi, J. S., Chiroma, H., Abdulhamid, S. M., Adetunmbi, A. O., & Ajibuwa, O. E. (2019). Machine learning for email spam filtering: Review, approaches, and open research problems. *Heliyon*, 5(6), e01802.

Dangeti, P., Kumar, A., & Das, S. (2024). Classification analysis for email spam using machine learning and feedforward neural network approaches. *International Journal of Computer Applications*, 176(3), 35–42.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 4171–4186.

Guzella, T. S., & Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7), 10206–10222.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mansoori, F., Deldari, H., & Rezaei, A. (2023). Email spam classification based on deep learning methods: A review. *Journal of Machine Intelligence and Data Science*, 6(1), 25–41.

Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006). Spam filtering with naive Bayes – Which naive Bayes? *Third Conference on Email and Anti-Spam (CEAS).*

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. *INTERSPEECH*, 1045–1048.

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. *Learning for Text Categorization: Papers from the 1998 Workshop*, 98(1), 55–62.

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune BERT for text classification? *China National Conference on Chinese Computational Linguistics*, 194–206.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N. & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998–6008.

Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of CNN and RNN for natural language processing. *arXiv preprint arXiv:1702.01923*.

Zhang, L., Zhu, J., & Yao, T. (2004). An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing (TALIP),* 3(4), 243–269.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, 28.