NOVEL DEEP LEARNING APPROACH FOR SECURING EMAIL COMMUNICATIONS AGAINST EMERGING CYBER THREATS

A PROJECT REPORT

Submitted by

AKASH V	**(715321104006)**
DHINESH S	**(715321104019)**
KISHORE KUMAR S	**(715321104027)**
VANMATHI P	**(715321104055)**

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

ASIAN COLLEGE OF ENGINEERING AND TECHNOLOGY

COIMBATORE

ANNA UNIVERSITY: CHENNAI 600 025

APRIL/MAY 2025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIED CERTIFICATE

Certified that this project report **" NOVEL DEEP LEARNING SECURING EMAIL COMMUNICATIONS AGAINST EMERGING CYBER THREATS"** is the Bonafide work of **" AKASH V (715321104006),DHINESH S (715321104019),KISHORE KUMAR S (715321104027),VANMATHI P (715321104055)"** Who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mr.G.VADIVEL,M.E.,	**Mr.G.VADIVEL,M.E.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Assistant professor, Department of CSE	Assistant professor, Department of CSE
Asian College of Engineering and	Asian College of Engineering and
Technology, Coimbatore-641 110.	Technology, Coimbatore-641 110.

Submitted for the project viva voce held on ____________________________

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

“Thanks” may be a little but its eloquence is magnified only when it is spelled from the depth of the heart. At this point of time, we would like to extend my heart full thanks to all those who helped me through this major assignment.

We place a record of deep sense of gratitude to our beloved chairman **Mr.A.SELVARAJ, B.Sc.,** for the keen interest and affection towards us throughout the course. We take the opportunity to extend our hearty thanks to our principal **Dr.P.ESHWARAMOORTHI, B.E.,M.E.,Ph.D.,** Asian College of Engineering and Technology for this interest towards us throughout the course.

I express our performed gratitude to **Mr.G.VADIVEL.M.E ,**and head of the department and to my project guide **Mr.G.VADIVEL.M.E,** for his unique innovation plans, dynamic guidance with constant encouragement and motivation, which trigger us great extent in completion

I also extend our sincere thank to **Mr.G.VADIVEL.M.E,** our project coordinator for his valuable support and help. I also thank to all my **faculty members** for volunt5eering their time to read my project work and providing feedback to improve it. Last but not least; I am greatly in debuted to beloved **Parents, Brothers, Sisters, and Friends** who helped us throughout this project.

KISHORE KUMAR S	**(715321104027)**
AKASH V	**(715321104006)**
DHINESH S	**(715321104019)**
VANMATHI P	**(715321104055)**

B.E COMPUTER SCINECE AND ENGINEERING

ABSTRACT

Email has become an essential communication tool, seamlessly facilitating information exchange across personal and professional spheres. While its convenience and global accessibility are unparalleled, email systems have increasingly become targets for cybercriminals exploiting sophisticated spam tactics to breach government networks, corporate systems, and individual accounts. These threats are characterized by their complexity and scale, outpacing traditional detection mechanisms and emphasizing the need for innovative and adaptive solutions to combat emerging cyber risks effectively. This project proposes an advanced system for classifying large-scale email datasets into four distinct categories: Normal, Fraudulent, Harassment, and Suspicious. The approach integrates Natural Language Processing (NLP) with Bidirectional Long Short-Term Memory (BiLSTM) networks to capture nuanced patterns and semantic meanings within email content. The methodology includes a sample expansion phase to enhance training data diversity and a robust testing stage to ensure high accuracy under varied conditions. This innovative system enables effective forensic analysis by extracting and analysing meaningful information from email communications. Through extensive experimentation, the proposed system demonstrates a significant improvement over existing machine learning techniques, achieving a remarkable classification accuracy of 99.1%. The use of BiLSTM with recurrent gradient units ensures reliable performance across diverse email topics and complex scenarios. By offering a highly accurate and robust solution, this project contributes to advancing email security .

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATION

S.NO	ABBREVATION	EXPANSION
1	AI	Artificial Intelligence
2	KYC	Know Your Customer
3	ML	Machine Learning
4	LSTM	Long Short-Term Memory
5	CNN	Cable News Network
6	RNN	Recurrent Neural Network
7	CSV	Comma-Separated Values
8	KNN	K-Nearest Neighbors

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Email stands for Electronic Mail. It is a method to sends messages from one computer to another computer through the internet. It is mostly used in business, education, technical communication, document interactions. It allows communicating with people all over the world without bothering them. In 1971, a test email sent Ray Tomlinson to himself containing text.

Fig 1.1. E-Mail

Email messages are conveyed through email servers; it uses multiple protocols within the TCP/IP suite. For example, SMTP is a protocol, stands for simple mail transfer protocol and used to send messages whereas other protocols IMAP or POP are used to retrieve messages from a mail server. If you want to login to your mail account, you just need to enter a valid email address, password, and the mail servers used to send and receive messages.Although most of the webmail servers automatically configure your mail account, therefore, you only required to enter your email address and password. However, you may need to manually configure each account if you use an email client like Microsoft Outlook or Apple Mail. In addition, to enter the email address and password, you may also need to enter incoming and outgoing mail servers and the correct port numbers for each one.

Email messages include three components, which are as follows:

- Message envelope: It depicts the email's electronic format.
- Message header: It contains email subject line and sender/recipient information.

1.2 PROBLEMS DEFINITION

E-mail is an essential application for carrying out transactions and efficiency in business processes to improve productivity. As one of the most popular services, email has become a major vulnerability to users and organizations. E-mail is frequently used as a vital medium of communication and is also being used by cybercriminals to commit crimes. The statistics are astounding. Email remains the number one threat vector for data breaches, the point of entry for ninety-four percent of breaches. There is an attack
every 39 seconds. Over 30% of phishing messages get opened, and 12% of users click on malicious links. As cybercrime becomes more advanced and bypasses the legacy controls put in place to defend against it, security must become more advanced too. Cybercrimes like hacking, spoofing, phishing, E-mail bombing, whaling, and spamming are being performed through E-mails. Hence, there is a need for proactive data analysis to prevent cyber-attacks and crimes. To investigate crimes involving Electronic Mail (e-mail), analysis of both the header and the email body is required since the semantics of communication helps to identify the source of potential evidence. With the continued growth of data shared via emails, investigators now face the daunting challenge of extracting the required semantic information from the bulks of emails, thereby causing a delay in the investigation process. The existing email classification approaches lead towards irrelevant E-mails and/or loss of valuable information.

1.3 AI AND CYBER SECURITY

The cyberattack surface in modern enterprise environments is massive, and it’s continuing to grow rapidly. This means that analysing and improving an organization’s cybersecurity posture needs more than mere human intervention. AI and machine learning are now becoming essential to information security, as these technologies are capable of swiftly analysing millions of data sets and tracking down a wide variety of cyber threats — from malware menaces to shady behaviour that might result in a phishing attack.

1.3.1 ML AND DL ALGORITHMS FOR ENABLING ARTIFICIAL INTELLIGENCE CYBERSECURITY

AI cybersecurity, with the support of machine learning, is set to be a powerful tool in the looming future. As with other industries, human interaction has long been essential and irreplaceable in security. While cybersecurity currently relies heavily on human input, we are gradually seeing technology become better at specific tasks than we are.

Artificial intelligence (AI) is designed to give computers the full responsive ability of the human mind. This is the umbrella discipline under which many others fall, including machine learning and deep learning.

Machine learning (ML) uses existing behaviour patterns, forming decision-making based on past data and conclusions. Human intervention is still needed for some changes. Machine learning is likely the most relevant AI cybersecurity discipline to date.

Deep learning (DL) works similarly to machine learning by making decisions from past patterns but makes adjustments on its own. Deep learning in cybersecurity currently falls within the scope of machine learning, so we'll focus mostly on ML here.

1.3.2 NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding.

NLP Techniques

Natural Language Processing (NLP) applies two techniques to help computers understand text: syntactic analysis and semantic analysis.

Syntactic Analysis

Syntactic analysis – or parsing – analyzes text using basic grammar rules to identify sentence structure, how words are organized.

Some of its main sub-tasks include:

- **Tokenization** consists of breaking up a text into smaller parts called *tokens* (which can be sentences or words) to make text easier to handle.
- **Part of speech tagging (PoS tagging)** labels tokens as *verb, adverb, adjective, noun,* etc. This helps infer the meaning of a word (for example, the word "book" means different things if used as a verb or a noun).
- **Lemmatization & stemming** consist of reducing inflected words to their base form to make them easier to analyze.
- **Stop-word removal** removes frequently occuring words that don't add any semantic value, such as *I, they, have, like, yours,* etc.

1.3.3 BI-DIRECTIONAL LSTM

A bidirectional LSTM, often known as a biLSTM, is a sequence processing model that consists of two LSTMs, the first model takes the input as it is, and the second model takes a backward direction copy of the sequence. Bidirectional long-short term memory(bi-lstm) is the process of making any neural network o have the sequence information in both directions backwards (future to past) or forward (past to future).

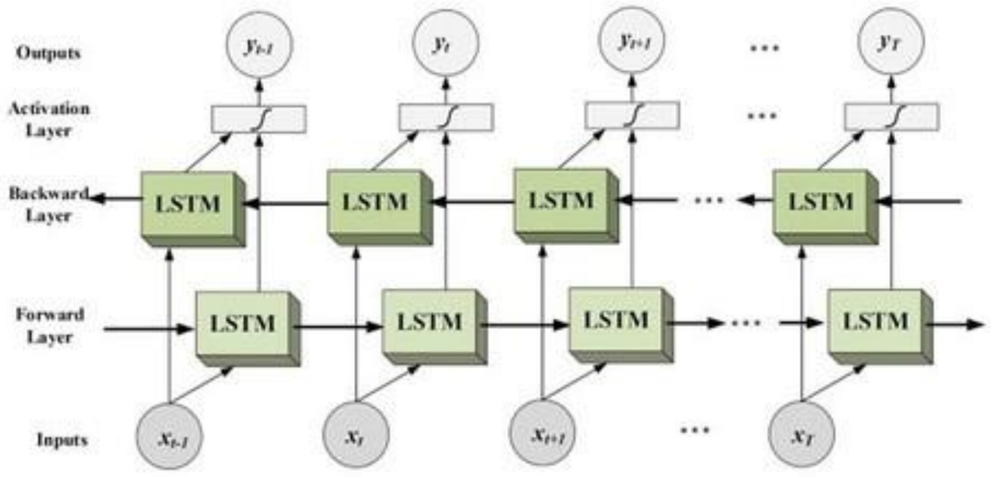


Fig 1.3.3 LSTM

AIM AND OBJECTIVE In the diagram, one can see the flow of information from backward and forward layers. BI-LSTM is usually employed where the sequence to sequence tasks are needed. This kind of network can be used in text classification.

1.4 AIM

The primary aim of the project is to develop an advanced AI-based system for the automatic classification and detection of spam emails in both private and public email services. The system utilizes Natural Language Processing (NLP) techniques and Bidirectional Long Short-Term Memory networks (BiLSTM) to enhance the accuracy of spam detection.

Objective

- To develop an advanced AI-based system for spam email classification using NLP and BiLSTM.
- To classify emails using NLP techniques and BiLSTM into Normal, Fraudulent, Harassment, and Suspicious.
- To integrate with private and public email services through a seamless API.
- To ensure real-time email analysis upon arrival for instant spam protection.
- To enable automatic deletion of classified spam emails.
- To implement an auto-reply system informing users about detected spam and deletion.
- To curate a diverse dataset encompassing various spam types for model training.
- To evaluate model performance using accuracy, precision, recall, and F1 score.
- To ensure scalability and reliability of the email screener as user demand grows.
- To enhance security by applying measures against potential vulnerabilities.

1.5 SCOPE OF THE PROJECT

- The scope of the project encompasses the development of an advanced spam detection and classification system using NLP and BiLSTM, with a focus on real-time processing, integration with email services, user feedback, scalability, security, and continuous improvement.

- **Spam Mail Detection:** The primary scope of the project is to develop a robust system capable of detecting spam mails in real-time. Leveraging Natural Language Processing (NLP) and Bidirectional Long Short-Term Memory (BiLSTM).

- **Classification of Spam:** The project aims to classify spam mails into different categories based on their content and characteristics. This includes identifying fraudulent emails, harassing messages, suspicious communications, and other forms of unwanted content.
- **NLP Preprocessing:** Implementing NLP techniques for preprocessing email content, including text normalization, tokenization, and feature extraction. This ensures that the input data is effectively prepared for analysis by the BiLSTM model.
- **BiLSTM Model Development:** Designing and training a BiLSTM neural network model to learn patterns and dependencies in email text data. The model will be trained on a diverse dataset of both
- **Real-Time Processing:** Implementing mechanisms for real-time processing of incoming emails to enable prompt detection and classification of spam
- **Integration with Email Services:** The project will include the development of APIs or plugins that seamlessly integrate with existing email services, both private and public.
- **Auto-Deletion and Notification:** Implementing functionality to automatically delete detected spam mails users inboxes.

CHAPTER 2

LITERATURE SURVEY

2.1 TRAINING NEURAL NETWORKS BY ENHANCE GRASSHOPPER OPTIMIZATION ALGORITHM FOR SPAM DETECTION SYSTEM

Author: Sanaa A. A. Ghaleb

Year:2021

Link: https://ieeexplore.ieee.org/document/9516002

Objective

This article proposes a new Spam Detection System (SDS) framework, by using a series of six different variants of enhanced Grasshopper Optimization Algorithm (EGOAs), which are investigated and combined with a Multilayer Perceptron (MLP) for the purpose of advanced spam email detection.

Methodology

The methodology of this study designs and implements the SDS model based on an ANN. The purpose is to design a new SDS model that achieves promising scores in terms of classification accuracy, detection rate, false alarm rate, global convergence and exhibits strong robustness in identifying spam emails with the help of a series of six different variants of enhanced Grasshopper Optimization Algorithm (EGOAs) for training the ANNs. MLPs are powerful classification tools that have proven highly capable of solving the challenges faced by SDS. **Dataset**

To evaluate the performance of the developed model, the author used three datasets composed of sets of email spam messages. The sets of messages are marked as spam or non-spam. These datasets are commonly used in evaluating spam detection systems. The SpamBase dataset is one of the prominently ancient datasets, which is still being used in the current research. The Spam Assassin dataset is a free, open-source, flexible, and powerful spam-fighting tool where it is characterized by removing duplicate and irrelevant data.

Findings

However, this research only evaluated the models by using all the features of the spam detection datasets where an adequate feature selection technique has not been included. Therefore, in the future our research will focus on developing an effective SDS based on two goals: the first is to reduce the number of selected features, and the second is to hybridize among the best-proposed models to build an ensemble-based classifier for spam detection

2.2 DECISION TREE MODEL FOR EMAIL CLASSIFICATION

Author: Ivana Čavor

Year: 2021

Link: https://ieeexplore.ieee.org/document/9390143

Objective

The aim is to preserve the most important features and to reduce computations demand. After feature selection, the ID3 algorithm is used to generate a decision tree that categorizes emails as spam or ham. The proposed approach is evaluated using accuracy, precision and recall. The performance of proposed system is measured against the size of dataset and feature size.

Methodology

This section presents proposed Spam Detection (SD) system in detail. The system goes through two stages: training and testing. The training stage has four modules: Data preparation, Feature selection, Feature reduction and Classification. The testing stage consist of Data preparation and Classification modules. A decision tree uses a tree-like model to represent a number of possible decision paths as well as their potential outcomes. Each decisions tree node represents a feature, each branch represents a decision and each leaf represents an outcome (class or decision). Decision trees can be used to predict the class of an unknown query instance by building a model trained on set of labelled data. Each training example should be characterized by a number of descriptive features or attributes. The features can have either nominal or continuous values.

Dataset

The dataset used for the classification purpose consists of 4000 entries. The dataset contains 3465 ham and 535 spam messages. This dataset is divided into two subsets: training set and testing set. The size of dataset assigned for training purpose can affect systems performance which will be shown further on.

Findings

For a classifier, accuracy is the proportion of the total testing examples which classifier predicted correct, precision is ratio of total number of correctly classified spam emails and the total number of emails predicted as spam and recall represents proportion of emails correctly classified as spam among all spam emails. The performance of proposed SD system is measured against the size of dataset and the features size. In the near future, it precision is planned to incorporate other classifiers and to compare their performances with the proposed approach.

2.3 EMAIL EMBEDDINGS FOR PHISHING DETECTION

Author: Luis Felipe Gutiérrez

Year:2021

Link: https://ieeexplore.ieee.org/document/9377821

Objective

In this paper, the author crafted a set of phishing and legitimate emails with similar indicators in order to investigate whether these cues are captured or disregarded by email embeddings, i.e., vectorizations.

Methodology

Support Vector Machine: Support Vector Machines (SVMs) determine a hyperplane that maximizes the margin between itself and the nearest samples of each data class. Such nearest samples are known as support vectors. This construction can be performed in the original input space, or a feature space that is generated by applying a kernel function to each pair of samples, which allows the decision boundary to be non-linear. In our work, we used the linear, Radial Basis Function (RBF), polynomial, and sigmoid kernels. An SVM converges towards non-linearly separable classes using the C hyperparameter, where C controls the number of samples that can be misclassified, and its optimal value.

Dataset

Twenty-four email stimuli were created for an experiment with human subjects. The content of the emails was modelled off legitimate emails found in the authors' inboxes or online. For all emails, a sender address, subject line, email body, and URL were created. The body of all emails was 50-100 words long and contained a hyperlink. Because the email stimuli were created to later show to research participants, this method for creating emails and the characteristics of the email stimuli are similar to those used in other phishing experiments. Twelve of the 24 email stimuli were legitimate emails, and the other 12 emails were phishing. All legitimate emails shared two characteristics.

Findings

The overall high classification results suggest that the semantic vector space in which the document vectors are is appropriate for this classification task. Moreover, the semantics of the emails' content are well suited for the class segmentation. As future work, we will explore the use of features that permit an easier interpretation and provide a deeper insight into phishing and legitimate emails. There are some other intriguing approaches to address the phishing email detection problem.

2.4 PHISHING ATTACK DETECTION USING MACHINE LEARNING CLASSIFICATION TECHNIQUES

Author: Tawsif Sarwar

Year:2020

Link: https://ieeexplore.ieee.org/document/9315895

Objective

By the use of supervised machine learning methods, through analysing the URLs, website structure, and other feature differences between phishing websites and legitimate websites, proposed work aimed to predict whether a website is phishing or not. This study mainly focusses on classifying phishing websites and legitimate websites by using several supervised machine learning methods.

Methodology

Machine Learning is a study of algorithms were using mathematical modelling with probabilistic theories decision making for solving a problem is done based on some amount of previous data or scenario of that problem. Machine learning is building mathematical models, integration of high-level equations which output the value of a target variable based on some dependent variable. Analysing the data of phishing and legitimate websites, based on their different characteristics, a machine learning model can predict whether a new unknown website would be phishing or a legitimate one. Supervised learning is a predictive model built on known outcomes.

Dataset

The dataset is one of the most critical parts of our study. A dataset is nothing but the table containing information about phishing and legitimate websites—the dataset for our proposed model obtained from Kaggle. Kaggle is one of the most popular public repositories with a tremendous amount of dataset collection which can be used for training machine learning models.

Findings

In this paper, the performance of three widely used machine learning classifiers are compared. Among these three classifiers, random forest performance is the highest with a precision of 97%. The AUC of the random forest is 1.0, which means our system can detect phishing website a high accuracy. In future, the accuracy improvement task will be done by changing features.

2.5 MALICIOUS MAIL FILTERING AND TRACING SYSTEM BASED ON KNN AND IMPROVED LSTM ALGORITHM

Author: Da Xiao; Meiyi Jiang

Year:2020

Link: https://ieeexplore.ieee.org/document/9251164

Objective

This paper describes an email filtering system that can effectively deal with phishing email attacks by machine learning and deep learning methods. The system can not only distinguish the malicious emails (i.e., spam and phishing emails) from the normal emails, but also distinguish the source of phishing emails according to the similarity between the mails from the same attacker.

Methodology

In this system, KNN classifier is used to distinguish normal mails, spam mails and phishing mails. This process includes two steps. Firstly, the phishing mails are detected from the data set. Secondly, the term frequency–inverse document frequency (TF-IDF) value of the words in the body of the mail is calculated by using the method of text classification to distinguish the spam mails from the normal mails. Long short-term memory (LSTM) algorithm is very suitable for modelling temporal data, such as text data, because of its design characteristics.

Dataset

Spam and normal mails are from" trec06p", a public English corpus provided by TREC." Trec06p" includes 37822 emails, 12910 of which are normal emails and 24912 of which are spam emails. Because the dataset of phishing mail is difficult to obtain, and the number is relatively small, in order to maintain the balance of data quantity in the process of KNN classification, we randomly selected 1200 normal mails and 1200 spam mails from the" trec06p" dataset as the experimental dataset.

Findings

The result of classification is relatively good, which has the value of further research and application. In order to further put the system into application, will study and use the API provided by the existing mailbox service to directly access and process users' mails.

2.6 HYBRID WATER CYCLE OPTIMIZATION ALGORITHM WITH SIMULATED ANNEALING FOR SPAM E- MAIL DETECTION

Author: Rabiei Mamat

Year:2019

Link: https://ieeexplore.ieee.org/document/8850011

Objective

This study is focused on the dimensionality of spam email classifiers. Hence, the mechanism of feature selection can be a curse to the dimensionality of the selection of relevant features and its classification. However, with the elimination and the reduction of the redundant features, several features can be reduced, and the training time can be increased. This will improve the classification performance. This study discussed the various limitations of the popular algorithms used in previous studies of feature selection

Methodology

In this algorithm, evaporation is the major influencing event because it can prevent the algorithm from fast convergence. Naturally, water is lost from rivers and lakes via evaporation but during photosynthesis, water is released by plants. The water that is lost through evaporation processes, upon entering the atmosphere, forms clouds. Later the clouds, upon contact with the colder atmosphere, condenses and returns as rain. The rain accumulates into streams, and streams flow into rivers, and rivers into seas.

Dataset

The dataset comprised 57 attributes and 4601 emails, where 1813 of the emails were spam, leaving the remaining 2788 as regular emails. The employed dataset is multivariate, containing actual integer attributes.

Findings

Based on this finding, it can be concluded that the content classification performance will be improved with enhancements to WCA as a feature selection. The second finding is that the use of the interleaved hybridization generated better optimal features for the SVM classifier than using all the features from this observation, it can be stated that content classification can be better performed using all the optimal features generated by the interleaved hybridization of WCA with SA. Future work could further be used for further researches in several fields of a kind such Used deep learning and to use the Bag-of-Narratives instead of Bag-of-Words and the semantic of each word.

2.7 PHISHING EMAIL DETECTION USING IMPROVED RCNN MODEL WITH MULTILEVEL VECTORS AND ATTENTION MECHANISM

Author: Yong Fang; Cheng Zhang

Year:2019

Link: https://ieeexplore.ieee.org/document/8701426

Objective

In this paper, the author first analyzed the email structure. Then, based on an improved recurrent convolutional neural networks (RCNN) model with multilevel vectors and attention mechanism, then proposed a new phishing email detection model named THEMIS, which is used to model emails at the email header, the email body, the character level, and the word level simultaneously.

Methodology

In this paper, emails are divided into two categories, legitimate emails and phishing emails. Naturally, the detection for phishing emails is also a binary classification problem. The author mathematizes the problem and split an email into two parts, the header and the body. then define a binary variable y to represent the attributes of an email; that is, y = 1 means that the email is a phishing email and y = 0 means that the email is legitimate. In other words, y is the label of an email. The author follows the following steps to determine whether the email is a phishing email. To begin this process, we calculate the probability that the email is a phishing email, that is, P (y = 1).

Dataset

The experimental data come from the First Security and Privacy Analytics Anti-Phishing Shared Task (IWSPA-AP 2018). The email data sources of this dataset are diversified. The sources of the legitimate email include email collections from Wikileaks archives, such as the Democratic National Committee, Hacking Team, Sony emails, etc. There are also selected emails from the Enron Dataset and Spam Assassin.

Findings

The THEMIS model obtains a promising result. Several experiments are performed to demonstrate the benefits of the proposed THEMIS model. For future work, we will focus on how to improve our model for detecting phishing emails with no email header and only an email body.

2.8 PHISHLIMITER: A PHISHING DETECTION AND MITIGATION APPROACH USING SOFTWARE-DEFINED NETWORKING

Author: Tommy Chin

Year:2018

Link: https://ieeexplore.ieee.org/document/8387883

Objective

In this paper, the author present PhishLimiter, a new detection and mitigation approach, where we first propose a new technique for deep packet inspection (DPI) and then leverage it with software-defined networking (SDN) to identify phishing activities through e-mail and web-based communication. The proposed DPI approach consists of two components: phishing signature classification and real-time DPI.

Methodology

SDN is an emerging networking structure that aims to overcome limitations of legacy networks. The centralized management of SDN guarantees consistent policy enforcement, better scalability, holistic visibility and flex programmable network function. PhishLimiter leverages the programmability.

Dataset

This dataset contains numerous phishing threats including 11,055 categorized phishing websites based on a 30-feature classification model.

Findings

Evaluated the trustworthiness of each SDN flow to identify any potential hazards based on each deep packet inspection. Likewise, observed how the proposed inspection approach of two SF and FI modes within PhishLimiter detects and mitigates phishing attacks before reaching end users if the flow has been determined untrustworthy. Using our real-world experimental evaluation on GENI and phishing dataset, demonstrated that PhishLimiter is an effective and efficient solution to detect and mitigate phishing attacks with its accuracy of 98.39%.

2.9 EFFICIENT FEATURE SET FOR SPAM EMAIL FILTERING

Author: Reshma Varghese

Year:2017

Link: https://ieeexplore.ieee.org/document/7976886

Objective

Major contributions of our approach are (a) Find an efficient feature set for spam email filtering, (b) Analyzed the four kinds of features such as Bag-of-Word (BoW)s, Bigram Bagof-Word (BoW)s, PoS Tag and Bigram PoS Tag, (c) Studied the performance of a model on individual features as well as aggregation of optimal features, (d) Studied the performance a model on the classifiers such as AdaBoostJ48, Random Forest, and SMO.

Methodology

Feature Extraction is the process of extracting the features from the body of the emails. From the body part of the emails, extracted four different kinds of features. These features are named as Bag-of-Word (BoW)s, Bigram Bag-of-Word (BoW)s, PoS Tag and Bigram PoS Tag.

Dataset

The used dataset is Enron-Spam. This dataset consists of emails in their original form without any processing. From Enron-Spam dataset, we used 11794 emails. In which, 6854 emails are in Ham class and 4940 emails are included in Spam class. First, emails are partitioned into header part and body part.

Findings

Different categories of features were extracted from EnronSpam dataset to find the best feature set for spam email filtering. Mainly features from four categories, consisting of Bag-of-Word (BoW)s, Bigram Bag-of-Words, PoS Tag and Bigram PoS Tag features were used in this study. Rare features are eliminated using Naive Bayes score and features are selected based on Information Gain value. Feature occurrence matrix is constructed, which is weighted on Term Frequency Inverse Document Frequency (TF-IDF) values.

2.10 EMAIL SPAM FILTERING USING BPNN CLASSIFICATION ALGORITHM

Author: Simranjit Kaur Tuteja

Year:2016

Link: https://ieeexplore.ieee.org/document/7877720

Objective

The fundamental constitution of Neural Network consists of an input and output layers with hidden layers in between each of altering or stable number of neurons. The neurons function as simple computing elements to impersonate the biological signal propagation while reducing the experimental threat during the training stage.

Methodology

In this system, for detection of spam and phishing emails, back propagation along with multilayer feed forward artificial neural network has been used as a training algorithm. The author would also be implementing a k-mean clustering algorithm on the vector set in the pre-processing stage to increase efficiency. To detect spam emails using neural network, the two phases (training and testing) need to be done. The process of detecting spam and phishing emails using feed forward neural network. There are mainly three stages in the proposed model: pre-processing stage (including implementation of clustering algorithm), neural network training (using BPNN algorithm) and detection of spam and phish (using ANN feed forward). In this approach, 11 features have been implemented as a binary value (0 or 1) where value 1 would indicate appearance of the feature in the tested.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

1.Content-Based Filtering Technique

Algorithms analyse words, the occurrence of words, and the distribution of words and phrases inside the content of e-mails and segregate them into spam non-spam categories.

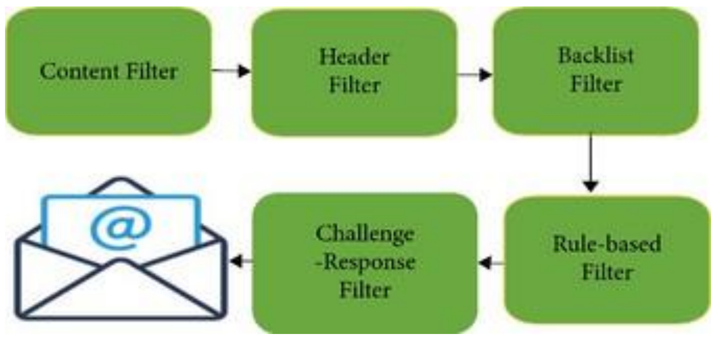


Fig 3.1 Content based Email Filtering

2.Case Base Spam Filtering Method

Algorithms trained on well-annotated spam/non-spam marked emails try to classify the incoming mails into two categories.

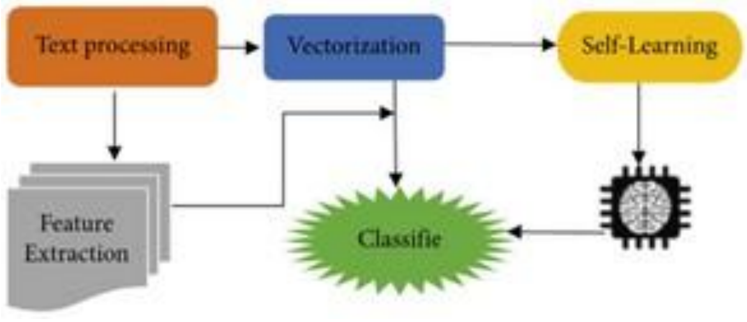


Fig 3.2 Case based Spam Filtering

3.Heuristic or Rule-Based Spam Filtering Technique

Algorithms use pre-defined rules in the form of a regular expression to give a score to the messages present in the e-mails. Based on the scores generated, they segregate emails into spam non-spam categories.

4.The Previous Likeness Based Spam Filtering Technique

Algorithms extract the incoming mails' features and create a multi-dimensional space vector and draw points for every new instance. Based on the KNN algorithm, these new points get assigned to the closest class of spam and non-spam.

5.Adaptive Spam Filtering Technique

Algorithms classify the incoming mails in various groups and, based on the comparison scores of every group with the defined set of groups, spam and non-spam emails got segregated.

6.Machine learning classifiers

The machine learning models are selected based on their group, diversity and acceptance in the machine learning community. SVM, Naive Bayes (NB) and DT are from three different groups of classifiers.

- **Support vector machine**

SVM are based on the assumption that the input data can be linearly separable in a geometric space. This is often not the case when working with real word data. To solve this problem SVM map the input to a high dimension feature space, i.e hyperplane, where a linear decision boundary is constructed in such a manner that the boundary maximises the margin between two classes. SVM is introduced as a binary classifier intended to separate two classes when obtaining the optimal hyperplane and decision boundary.

- **Decision tree**

A DT classifier is modelled as a tree where rules are learned from the data in a if-else form. Each rule is a node in the tree and each leaf is a class that will be assigned to the instance that fulfil all the above nodes conditions. For each leaf a decision chain can be created that often is easy to interpret. The interpretability is one of the strengths of the DT since it increases the understanding of why the classifier made a decision, which can be difficult to achieve with other classifiers. The telecommunication company is today using a manually created decision tree, in which the rules are based on different combinations of words.

- **Naive bayes**

The NB classifier is considered to perform optimal when the features are independent of each other, and close to optimal when the features are slightly dependant. Real world data does often not meet this criterion but researchers have shown that NB perform better or similar to C4.5, a decision tree algorithm in some settings. The researchers argue that NB performs well even when there is a clear dependency between the features, making it applicable in a wide range of tasks.

3.1.1 Disadvantage

- Predefined rule-based filtering lacks adaptability to evolving spam tactics, diminishing effectiveness over time.
- Content-based filtering suffers from limited feature extraction, missing subtle spam indicators
- Machine learning classifiers offer improved performance but introduce complexity and reduced interpretability.
- Classifier performance is heavily dependent on the quality and representativeness of training data.
- Machine learning classifiers are vulnerable to adversarial attacks, compromising system security

3.2 PROPOSED SYSTEM

The proposed approach comprises data collection, pre-processing, feature extraction, parameter tuning, and classification using the NLP and BiLSTM model. In this project, E-mail datasets are divided into normal, harassing, suspicious, and fraudulent classes. The E-mail is divided into word levels of the E-mail body, and the embedding layer is applied to train and obtain the sequence of vectors

- **Real-Time Spam Detection:** The system continuously monitors incoming email and applies machine learning algorithms to detect spam emails in real-time.
- **Multi-Class Classification:** Emails are classified into multiple categories including Normal, Fraudulent, Harassment, and Suspicious, allowing for more nuanced detection and handling of spam.
- **NLP-Based Analysis:** Natural Language Processing techniques are employed to analyze email content, extract meaningful features, and identify patterns indicative of spam or fraudulent activity.
- **BiLSTM Model:** The system utilizes Bidirectional Long Short-Term Memory (BiLSTM) networks, a type of recurrent neural network, to capture contextual information and improve the accuracy of email classification.
- **User-Friendly Interface:** The system features an intuitive user interface for both administrators and end-users, providing easy access to spam detection results, notifications, and configuration settings.
- **Automated Deletion and Alerting:** Detected spam emails are automatically deleted from the user's inbox, reducing the risk of user exposure to malicious content. Users also receive notifications/alerts via email about the detected spam mail and deletion actions.
- **Customizable Settings:** Administrators have the ability to customize system settings, including model parasmeters, data sources, and notification preferences, to suit the specific requirements of their organization.
- **Scalable Architecture:** The system is designed with scalability in mind, allowing for seamless integration with existing email servers and infrastructure, and supporting high volumes of email traffic.

3.2.1 ADVANTAGE

- Enhanced email security through real-time spam detection.
- Reduction in the risk of users falling victim to phishing scams or fraudulent activities.
- Time and resource savings with automated spam detection .
- Improved user experience with hassle-free email management.
- Customizable settings and adaptable architecture to meet diverse organizational needs.

- Seamless integration with existing email servers and infrastructure.
- Higher accuracy in identifying and filtering out spam emails using advanced technologies.
- Increased productivity by freeing users from manual handling of spam.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

● **Processor**	:	Intel Core i5 or higher (or equivalent AMD processor)
● **RAM**	:	8 GB or higher
● **Storage**	:	Minimum 256 GB SSD or HDD
● **Network Interface**	:	Ethernet or Wi-Fi connectivity
● **Display**	:	Minimum 13-inch display with a resolution of 1920x1080 pixels

4.2 SOFTWARE SPECIFICATION

● **Operating System**	:	Windows 10
● **Programming**	:	Python 3.8
● **Web Server**	:	WampServer (for local development and testing)
● **Web Framework**	:	Flask
● **Database**	:	MySQL
● **Frameworks**	:	TensorFlow and Keras
● **Libraries**	:	Matplotlib, Seaborn, and WordCloud
● **NLTK**	:	Natural Language Processing (NLP) Library

CHAPTER 5

SYSTEM DESIGN

5.1 DATA FLOW DIAGRAM

LEVEL 0

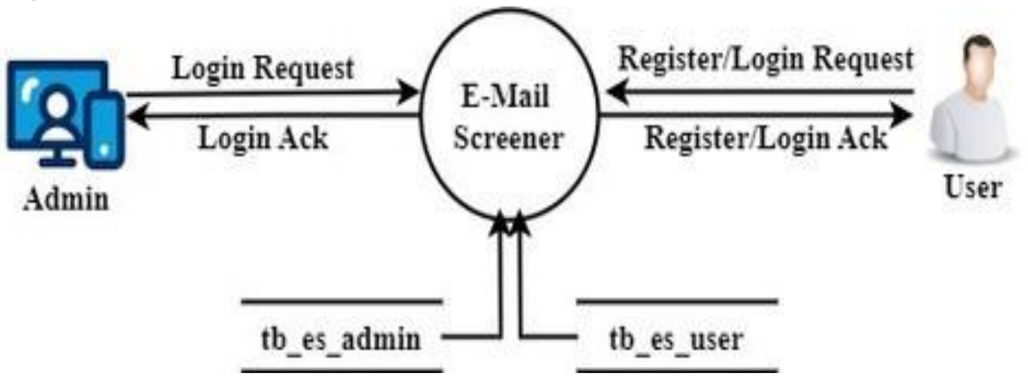


Fig 5.1 Level 0

LEVEL 1

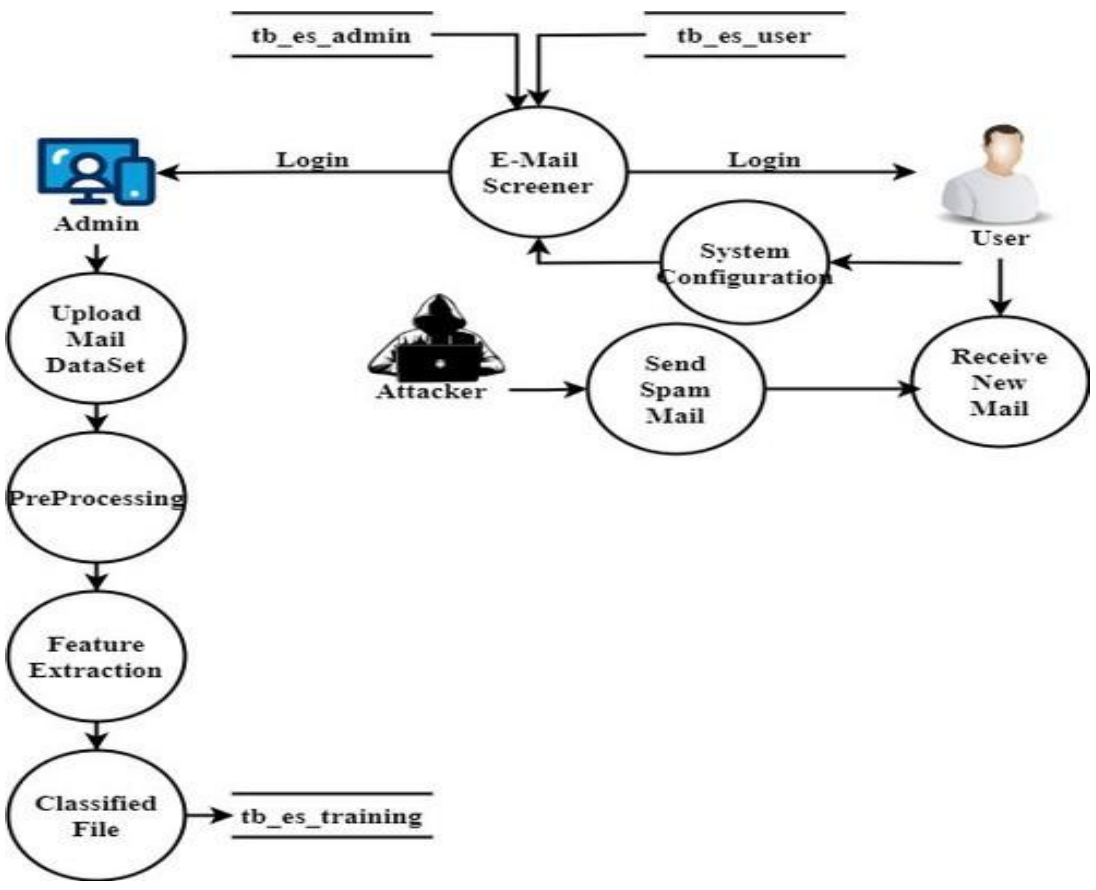


Fig 5.2 Level 1

LEVEL 2

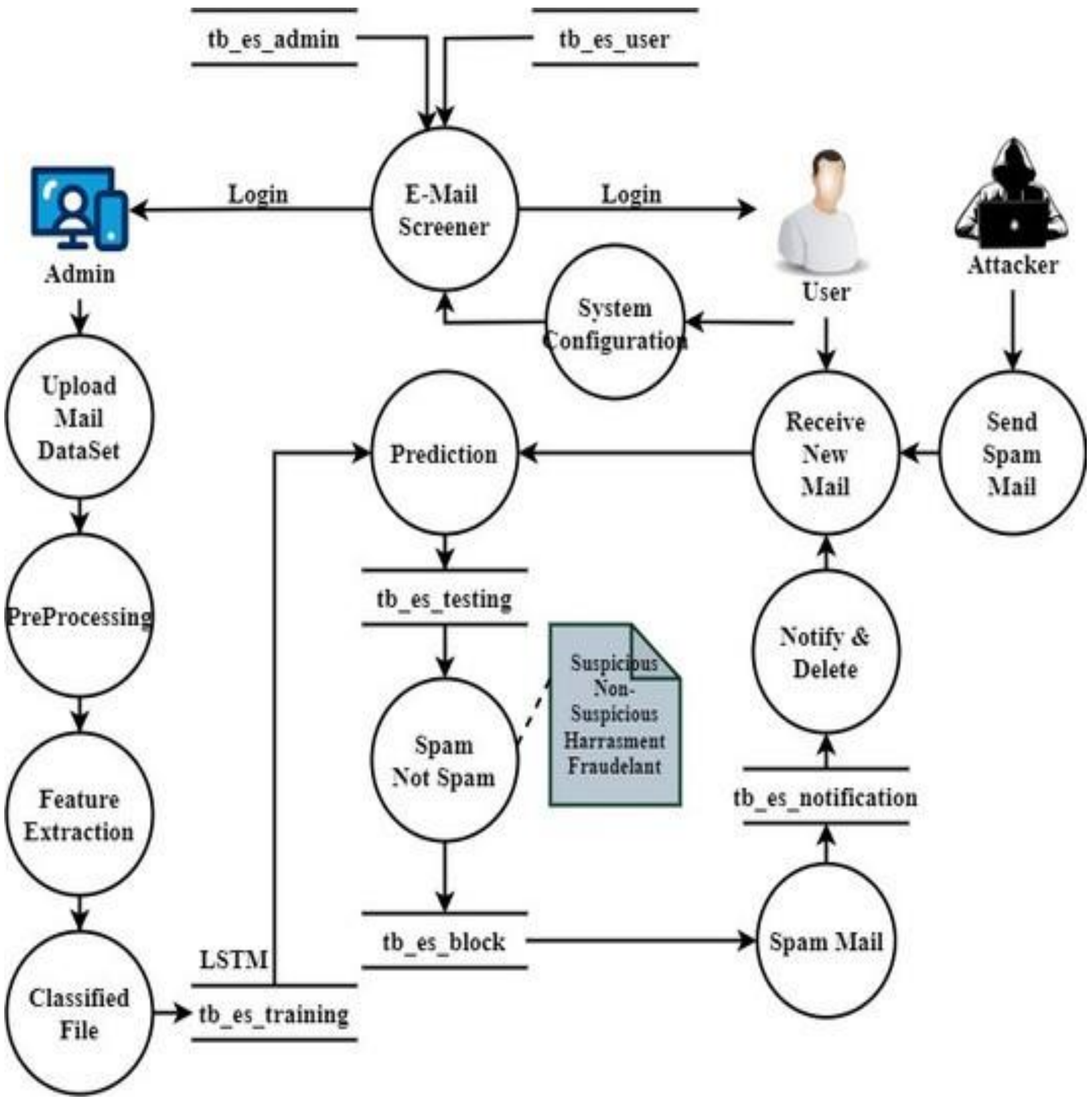


Fig 5.3 Level 2

5.2 UML DIAGRAM

5.2.1 USE CASE DIAGRAM

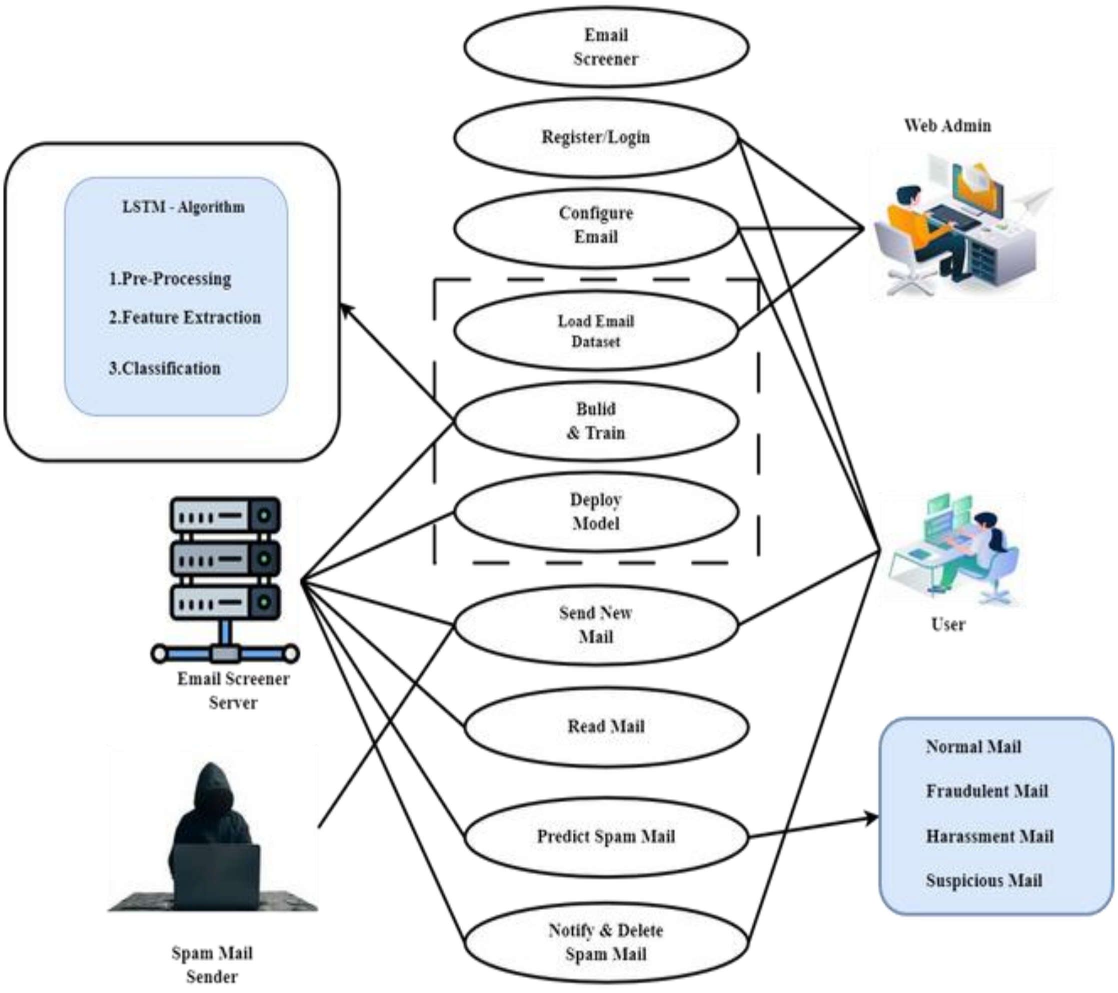


Fig 5.2.1 Use Case Diagram

5.2.2 ACTIVITY DIAGRAM

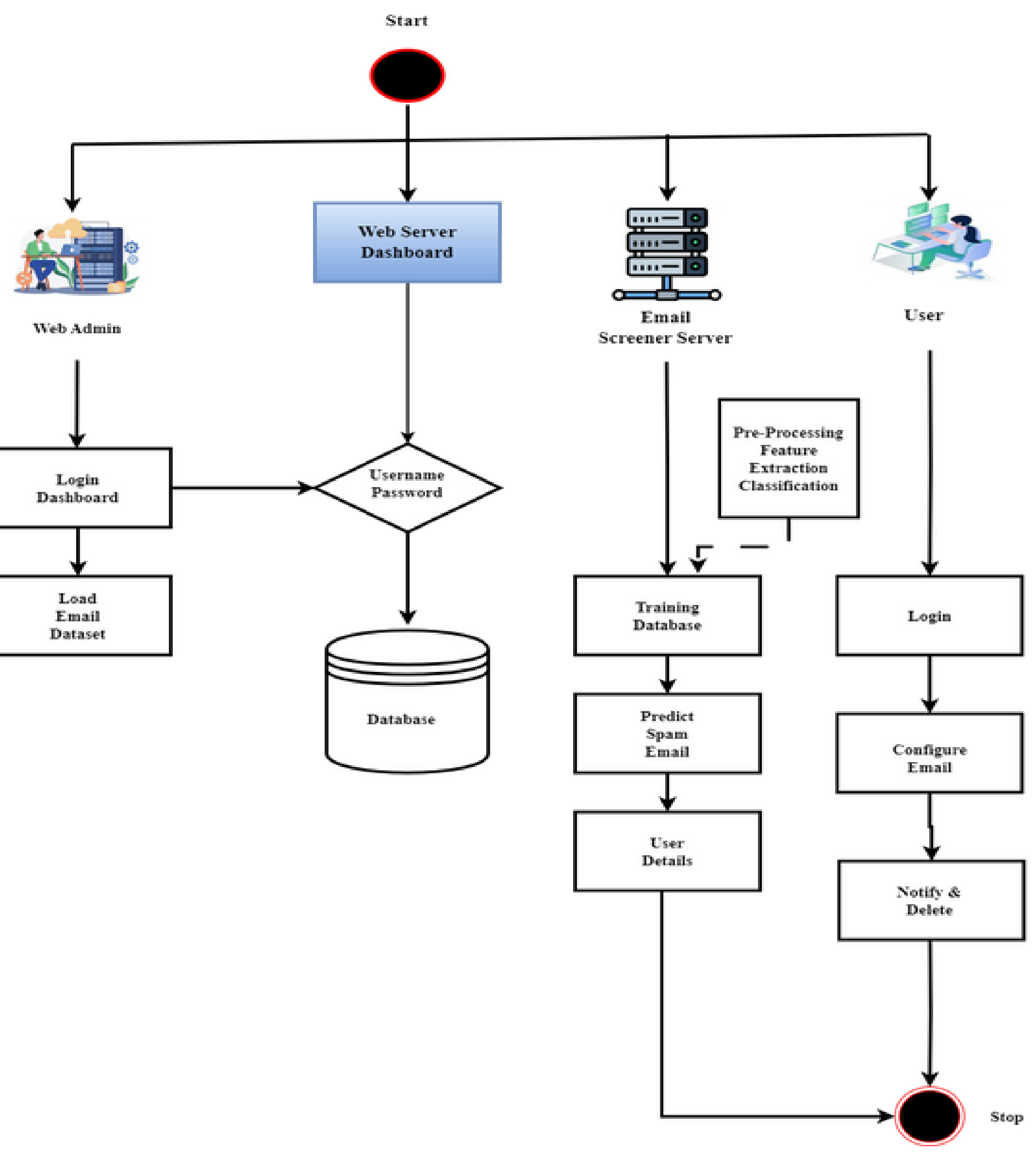


Fig 5.2.2 Activity Diagram

5.2.3 SEQUENCE DIAGRAM

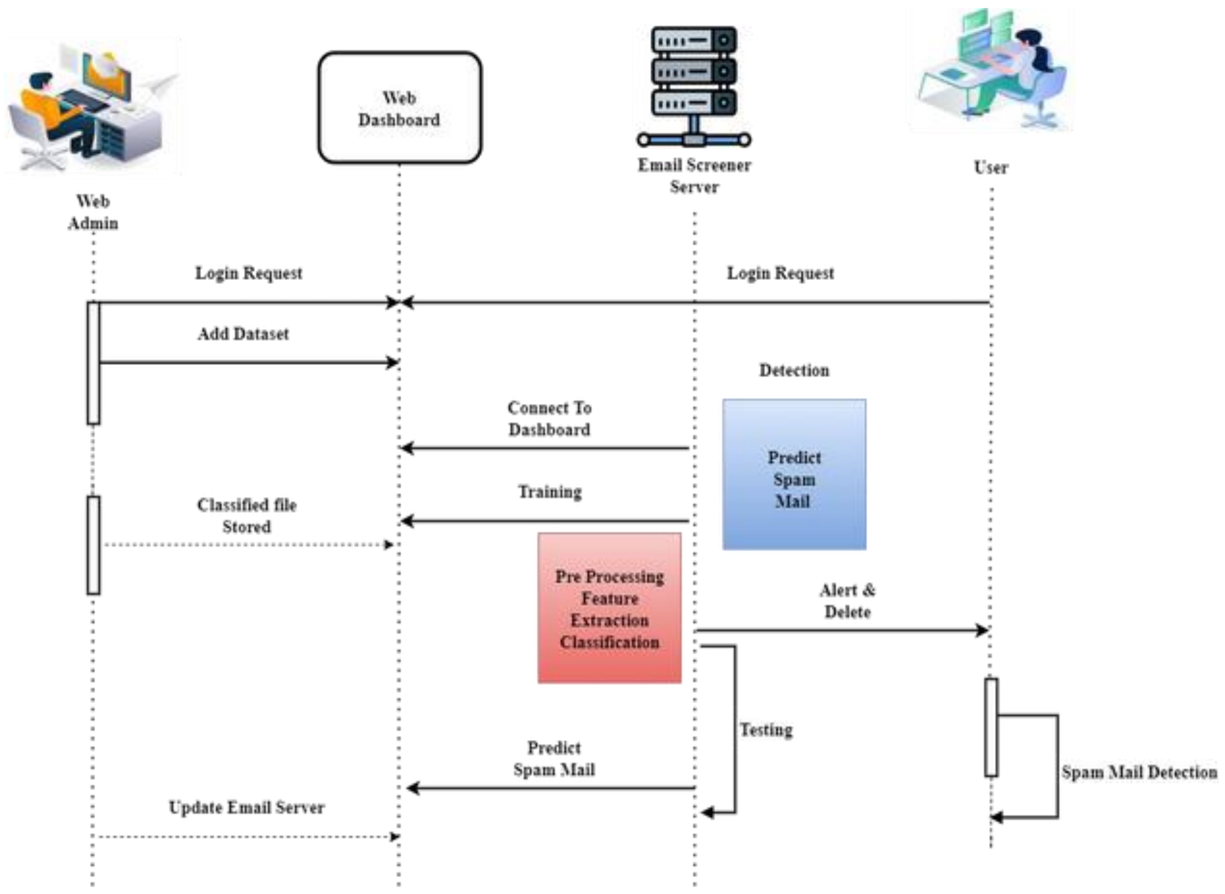


Fig 5.2.3 Sequence Diagram

5.3 TABLE DESIGN

1. TABLE NAME: ADMIN

S.No	Field	Data Type	Field Size	Constraint	Description
1	Username	Varchar	20	Null	Admin name
2	Password	Varchar	20	Null	Admin Password

Table 1

2. TABLE NAME: TRAINING

S.No	Field	Data Type	Field Size	Constraint	Description
1	Id	Int	11	Primary key	Id
2	Dataset link	Varchar	100	Null	Dataset link
3	Model Path	Varchar	100	Null	Email Screener model path

Table 2

3. TABLE NAME: USER REGISTER

S.No	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	Name	Varchar	20	Null	Name
3	City	Varchar	20	Null	City
4	Mobile	Bigint	20	Null	Mobile
5	Email	Varchar	20	Null	Email
6	Username	Varchar	20	Null	User name
7	Password	Varchar	20	Null	Password
8	Register Date	Timestamp	Timestamp	Primary key	Register Date

Table 3

4.TABLE NAME: TESTING

S No	Field	Data Type	Field Size	Constraint	Description
1	Id	Int	11	Null	Id
2	User Name	Varchar	20	Null	User Name
3	Predicted lable	Varchar	100	Null	Predicted lable
4	Spam mail Alert	Varchar	20	Null	Spam mail Alert
5	Date Time	Timestamp	Timestamp	Null	Date Time

Table 4

5.TABLE NAME: BLOCK SPAM MAIL DETAILS

S.No	Field	Data Type	Field Size	Constraint	Description
1	Id	Int	11	Null	Id
2	User Name	Varchar	20	Null	User Name
3	Subject	Varchar	100	Null	Subject
4	Message	Varchar	20	Null	Message
5	Block Spam Mail	Varchar	50	Null	Block Spam Mail

Table 5

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 PROJECT DESCRIPTION

The project is a sophisticated deep learning model designed to address the challenges associated with multiclass email classification for forensic analysis. With the pervasive use of email in both personal and professional communication, there has been a parallel increase in cybercrimes perpetrated through this medium. These crimes range from relatively benign spamming to more severe offenses like phishing, hacking, and threatening communications.

Traditional methods of email analysis often struggle to efficiently extract relevant semantic information from the vast volumes of email data. This inefficiency can lead to delays in forensic investigations and may result in the loss of valuable evidence. Project aims to overcome these limitations by leveraging advanced deep learning techniques. The architecture of project is based on Long Short-Term Memory (LSTM) cells with Gated Recurrent Units (GRU), which are well-suited for processing sequential data like natural language. LSTM networks are particularly effective at capturing long-term dependencies in sequences, making them ideal for analysing the complex structure of email content. The model's training process involves feeding it with labelled email datasets containing examples of various email classes, such as Normal, Fraudulent, Threatening, and Suspicious. During training, the LSTM-GRU architecture learns to extract meaningful features from both the email headers and bodies, allowing it to accurately classify emails into the appropriate categories. One of the key advantages of E-mailSinkAI is its ability to handle the intricacies of email content, including the nuances of language, context, and structure. This capability enables it to discern subtle differences between legitimate communications and potentially malicious activities, enhancing the accuracy of forensic analysis. Furthermore, project is designed to be robust and reliable, capable of handling large volumes of email data with high efficiency. Its performance surpasses that of traditional email classification methods, reducing the incidence of false positives and false negatives. In practical terms, project can significantly expedite forensic investigations involving email evidence. By automating the process of email classification and analysis, it empowers investigators to identify relevant evidence more quickly and effectively, ultimately aiding in the detection and prevention of cybercrimes. Overall, project represents a cutting-edge solution for email classification in forensic analysis, combining the power of deep learning with the intricacies of email communication to deliver accurate and efficient results.

6.2 MODULES DESCRIPTION

6.2.1 E-MAIL SCREENER WEB APP

The E-Mail Screener Web App module serves as a vital interface, providing users with a centralized platform to interact with the sophisticated spam mail detection system driven by Natural Language Processing (NLP) and Bidirectional Long Short-Term Memory (BiLSTM) technologies. Through this intuitive interface, users gain access to a suite of powerful features designed to enhance email security and streamline their inbox management process. At the core of the module is its user-friendly design, which prioritizes simplicity and ease of use. Users can effortlessly navigate through the app's various sections, accessing key functionalities with just a few clicks. Whether it's reviewing spam detection results, adjusting filtering settings, or seeking assistance, users can accomplish tasks efficiently within the app. One of the module's key functionalities is its ability to provide users with real-time insights into spam detection outcomes. By displaying classification results directly within the app, users gain immediate visibility into which emails have been identified as spam and which ones are legitimate. This transparency enables users to take swift action, ensuring that spam emails are promptly addressed and mitigated. Furthermore, the module offers robust customization options, allowing users to tailor spam filtering settings to suit their specific preferences and needs. From setting confidence thresholds to defining rules for handling detected spam emails, users have full control over how the system processes incoming messages. This level of customization empowers users to fine-tune the spam detection process, optimizing it for their unique requirements. In addition to its core functionalities, the module also provides comprehensive support resources to assist users as they navigate the platform. Whether it's accessing FAQs, knowledge base articles, or reaching out to customer support, users can easily find the assistance they need to resolve any issues or inquiries they may encounter. Thus, the E-Mail Screener Web App module represents a powerful tool for users to enhance email security and streamline their inbox management process. With its intuitive design, real-time insights, customization options, and robust support resources, the module empowers users to take control of their email security with confidence and ease.

6.2.2 END USER

Admin Module

- **Login:** This module allows administrators to securely log in to the E-Mail Screener system using their credentials. Authentication ensures that only authorized personnel can access administrative functionalities.
- **Upload E-Mail Dataset:** Administrators can upload email datasets for training and fine-tuning the spam detection model. The module supports various data formats and provides validation mechanisms to ensure data integrity.
- **Build and Train the Model:** In this module, administrators can initiate the process of building and training the spam detection model using the uploaded email dataset. The system employs advanced machine learning algorithms, including NLP and BiLSTM, to train the model effectively.
- **User Management:** Administrators have the authority to manage user accounts within the system.

  This includes creating new user accounts, updating user information, resetting passwords, and deactivating or deleting user accounts as needed.

User Module

- **Register:** New users can register for an account by providing necessary information such as username, email address, and password. The registration process includes validation steps to ensure the integrity of user data.
- **Login:** Registered users can securely log in to their accounts using their credentials. Authentication mechanisms protect user accounts.
- **Configure Personal Email:** This module allows users to configure their personal email accounts with the E-Mail Screener system. Users provide their original email usernames and passwords, enabling the system to access and monitor their email activity for spam detection purposes.

- **Receive Spam Mail and Deletion Alert:** Users receive notifications/alerts via their personal email accounts whenever spam mail is detected. Additionally, users are notified when detected spam emails are automatically deleted from their inbox. These alerts help users stay informed about the status of their email security and take necessary actions if required.

These modules collectively facilitate seamless interaction between administrators and end users, enabling efficient management of email datasets, model training, user accounts, and spam detection functionalities within the E-Mail Screener system.

6.2.3 EMAILNET MODEL: BUILD AND TRAIN

6.2 3.1 DATASET ANNOTATION

In this project, E-mails are divided into normal, harassing, suspicious, and fraudulent classes. The E-mail is divided into word levels of the E-mail body, and the embedding layer is applied to train and obtain the sequence of vectors.

6.2.3.2 E-MAIL DATA SET PREPARATION AND EXPLORATION

- **Import Dataset**

In this module the admin uploads an email dataset (CSV) file. This will be used to train your email forensics anal

The EmailSinkAI reads email dataset to output the purpose or objective of the project.

- **Explore Dataset – EDA**

Data visualization tool that brings the entirety of data together into a striking and easy-to-follow view.

6.2.3.3 Data Pre-processing

The data pre-processing phase consists of natural language-based steps that standardize the text and prepare it for analysis.

- **Tokenization**

Breaking up the original text into component pieces is the tokenization step in natural language processing. There are predefined rules for tokenization of the documents into words. The tokenization step is performed in Python.

- **Normalization**

These are the steps needed for translating text from human language (English) to machine-readable format for further processing.

- **Stop Words Removal**

Words like ``a" and ``the" that appear so frequently are not relevant to the context of the E-mail and create noise in the text data. These words are called stop words, and they can be filtered from the text to be processed. We utilized the``NLTK" Python library to remove stop words from the text.

- **Punctuation Removal**

Punctuation includes (e.g., full stop (.), comma (,), brackets) to separate sentences and clarify meaning. For punctuation removal, we utilize the ``NLTK" library.

6.2.3.4 FEATURE EXTRACTION

After eliminating irrelevant information, the elaborated list of words is converted into numbers. The TF- IDF method is applied to accomplish this task. Term Frequency is several occurrences of a word in a document, and IDF is the ratio of a total number of documents and the number of documents containing the term. A popular and straightforward method of feature extraction with text data is called the bag-of- words model of text. A bag-of-words model, or BoW for short, is a way of extracting features from the text for use in modelling, such as machine learning algorithms. A bag-of-words is a representation of text that describes the

$$
\begin{aligned}
TFIDF &= tf * \left(\frac{1}{df}\right) \\
TFIDF &= tf * Inverse(df) \\
TFIDF(t,d,D) &= TF(t,d).IDF(t,D) \\
TFIDF(t,d) &= log\frac{N}{|d \epsilon D t \epsilon D|}
\end{aligned}
$$

occurrence of words within a document. It involves two things (1)

Eq. 1-5

A vocabulary of known words, (2) A measure of the presence of known words. We extract features on the basis of Equations Here tf represents term frequency.

Feature extraction in DL with the context of words is also essential. The technique used for this purpose is word2vec neural network-based algorithm. Equation 5 given below shows how word2vec manages the word-context with the help of probability measures. The D represents the pair-wise illustration of a set of words, and (w; c) is the word-context pair drawn from the large set D.

$$P(D = 1 \mid w, c_{1:k}) = \frac{1}{1 + e^{-(w \cdot c_1 + w \cdot c_2 + \ldots + w \cdot c_k)}}$$

Eq.5

The multi-word context is also a variant of word2vec, as shown in Equation 6. The variable-length context is also controlled by the given below mathematics.

$$P(D = 1 \mid w, c) = \frac{1}{1 + e^{-s(w,c)}}$$

Eq.6

6.2.3.5 BILSTM CLASSIFICATION MODEL

BiLSTM comprises the classification of E-mails as Normal, Harassing, Suspicious, and Fraudulent.The DL models' layered structure helps in learning without intervention in ML model implementation. Several libraries provide an in-depth learning implementation structure. We split the data into three training,validation, and testing sets with a 65 V 10 V 25 ratio. We extracted the features from textual data of E-mail using the word Embedding technique. We encode the target values using the one-hot encoding technique into 4-distinct classes. We pass all pre-processed data to the novel architecture of BiLSTM layers

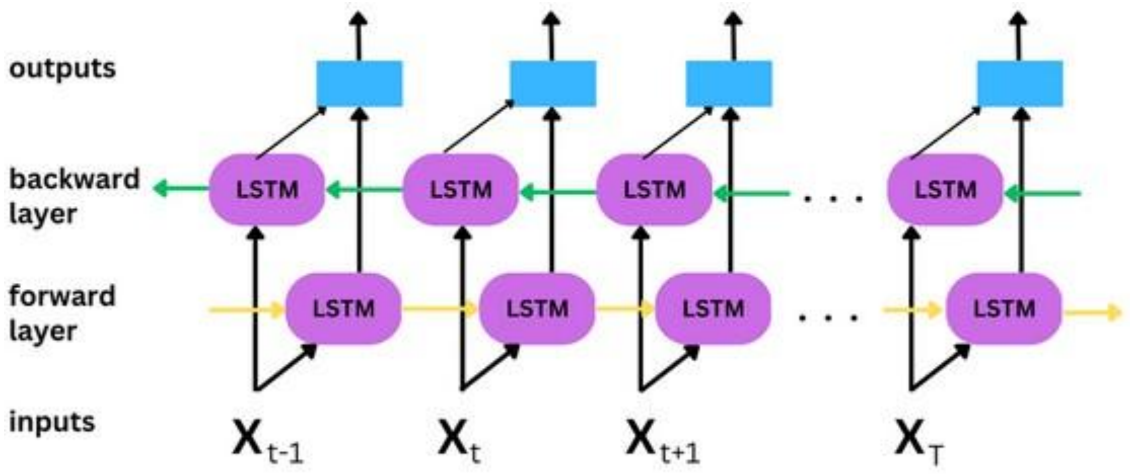


Fig 6.2.3.5 BiLSTM

variants for the perfect classification of E-mails. We use the BiLSTM layers with different GRU and Convo1D layer variants to transform the input textual data into an efficient E-mail classification system.

Textual data needs special attention when feature extraction comes in the proposed methodology. Different feature extraction methods need to be implemented when solving the Natural language processing problem using DL. The main point is to convert the textual data into real-valued vectors. There is a unique name for the vector in natural language processing, ``embedding vector''. There are multiple ways to generate the embedding vector from the textual data, but famous methods are GLOVE and Word2Vec techniques. Embedding vector dimensions are essential to get all the features extracted from the data. Let us suppose if we have 8 samples of textual data. The data have two distinct classes. Each sample has a maximum of five tokens in it. The vocabulary size will be the unique words in 8 samples, and the vocabulary size needs to be higher than the available unique tokens in the dataset to avoid collisions with a hash function. In this case, the dimensions of the embedding vector will be 4 x 8. In the case of the classification problem of NLP, we need to encode the target values using the one-hot encoding method.
After getting the vectors from the words, the similarity between the words is measured using the similarity measure between the corresponding vectors using Equation 7.

$$\mathrm{sim}_{\cos}(u, v) = \frac{u \cdot v}{\|u\|_2 \|v\|_2} = \frac{\sum_i u_{[i]} \cdot v_{[i]}}{\sqrt{\sum_i (u_{[i]})^2} \sqrt{\sum_i (v_{[i]})^2}}$$

Eq.7

There are many other ways to measure the similarity between the word vectors, one of them is Jaccard similarity, which defines Equation 8.

$$\mathrm{sim}_{\mathrm{Jacard}}(u, v) = \frac{\sum_i \min(u_{[i]}, v_{[i]})}{\sum_i \max(u_{[i]}, v_{[i]})}$$

Eq.8

DL for NLP uses dense vector representation to reduce the memory requirement for large models. Dense vector categorical data encoding is also a famous method, but most literature is based on one-hot encoding techniques. After feature extraction, the language modelling phase comes up.

6.2.3.6 BUILD AND TRAIN THE MODEL

The EmailNet model is built and trained using the preprocessed email dataset and the BiLSTM architecture. Training involves optimizing the model parameters (e.g., weights and biases) using techniques such as backpropagation and gradient descent to minimize classification errors. The training process iterates over multiple epochs, gradually improving the model's performance.

6.2.3.7 DEPLOY THE MODEL

Once the EmailNet model is trained and evaluated for performance, it is deployed within the E-Mail Screener Web App environment. The deployed model is integrated seamlessly into the app's backend infrastructure, enabling real-time spam detection and classification for incoming emails. The deployed model continuously monitors users' email inboxes, classifying incoming emails as spam or legitimate and providing users with accurate spam detection results within the E-Mail Screener Web App interface.

6.2.4 E-MAIL PREDICTION

This module of the E-Mail Screener system is responsible for predicting whether incoming emails are spam or legitimate, notifying users of detected spam, and automatically deleting spam emails from their inbox. The process involves real-time monitoring of users' personal email accounts, extraction of new emails, classification of emails as spam or normal, and sending notifications to users regarding detected
spam. Below is a detailed description of the components and functionalities of the E-Mail Prediction module:

6.2.4.1 EMAIL MONITORING AND EXTRACTION

The E-Mail Prediction module continuously monitors users' personal email accounts for new incoming emails. Upon detection of new emails, the module extracts relevant information such as sender, subject, body, and attachments for further processing.

6.2.4.2 SPAM CLASSIFICATION

Extracted email data is processed through the spam detection model trained using NLP and BiLSTM algorithms. The model predicts whether each email is spam or normal based on learned patterns and features extracted from the email content. Classification labels (e.g., "spam" or "normal") are assigned to each email based on the model's predictions.

6.2.4.3 NOTIFICATION OF DETECTED SPAM

If an email is classified as spam, the module generates a notification to alert the user about the detected spam. Notifications may include details such as the sender, subject.

6.2.4.4 AUTOMATIC DELETION OF SPAM EMAILS

Upon classification of an email as spam, the module automatically deletes the spam email from the user's inbox to prevent it from reaching the user. Deleted spam emails are permanently removed from the user's email account to ensure that they do not pose any further risk.

6.2.4.5 NOTIFICATION TO USER

Following the deletion of a spam email, the module sends a notification to the user's email address to inform them about the detected spam and the action taken. Notifications provide users with transparency regarding the spam detection process and help them stay informed about their email security.

6.2.5 NOTIFICATION

The Notification Module enables users to stay informed about their email security status and system updates through timely alerts. Users can configure their notification preferences, receiving notifications via email or other preferred channels. When a spam email is detected, users receive alerts detailing the detected spam and any actions taken, such as automatic deletion. Additionally, notifications about system updates and maintenance activities ensure users are aware of changes to the system's functionalities. With personalized settings and real-time delivery, the Notification Module enhances user engagement and awareness within the E-Mail Screener system.

CHAPTER 7

SYSTEM TESTING

7.1 SOFTWARE TESTING

Accuracy Testing: The system should be tested to ensure that it accurately classifies emails into the predefined categories of normal, harassing, suspicious, and fraudulent. This involves providing a diverse set of test emails representing each category and verifying that the LSTM-GRU model correctly classifies them with a high level of accuracy.

False Positive Rate Testing: Another important aspect of system testing is evaluating the false positive rate of the classification algorithm. This involves analyzing a set of legitimate emails (normal) and ensuring that the system does not misclassify them as harassing, suspicious, or fraudulent. A low false positive rate indicates that the system effectively distinguishes between legitimate and suspicious emails.

Efficiency Testing: The system should be tested for efficiency in processing email data, particularly in terms of computational resources and processing time. This involves measuring the time taken to classify a batch of emails and assessing the system's resource utilization (CPU, memory) during classification tasks. The goal is to ensure that the system can handle large volumes of email data efficiently without experiencing performance bottlenecks.

Adaptability Testing: The system should be evaluated for its ability to adapt to different types of email content and communication styles. This involves testing the system with diverse email datasets, including variations in language, writing style, and email structure. The system should demonstrate robust performance across various email categories, indicating its adaptability to real-world forensic analysis scenarios.

Scalability Testing: Scalability testing involves assessing the system's ability to handle increasing volumes of email data without compromising performance or accuracy. This can be done by gradually increasing the size of the email dataset used for testing and monitoring the system's classification performance. The system should demonstrate linear or near-linear scalability, indicating its ability to scale effectively with growing data volumes.

Generalization Testing: Generalization testing verifies the system's performance on unseen email datasets that were not used during training. This involves evaluating the system's classification accuracy and false positive rate on new email samples to assess its generalization capability. A well-generalized system should maintain high accuracy and low false positives across diverse email datasets.

Robustness Testing: Robustness testing assesses the system's resilience to noise, errors, and adversarial inputs. This involves subjecting the system to various forms of input perturbations, such as adding random noise to email content or modifying email headers. The system should demonstrate stable performance and maintain accuracy in the presence of such disturbances.

By conducting comprehensive system testing encompassing these aspects, the performance, reliability, and suitability of the LSTM-GRU model for multiclass email classification in forensic analysis can be thoroughly evaluated, ensuring its effectiveness in real-world deployment scenarios.

7.1.1 TESTING METHODOLOGIES

- **Unit Testing**

Objective: To test individual components of the system, such as data preprocessing, feature extraction, and classification algorithms, in isolation.

Scope: Focuses on verifying the correctness of each unit's functionality and ensuring that they behave as expected.

Approach: Utilizes techniques like test-driven development (TDD) to write test cases for each unit, including edge cases and boundary conditions.

Tools: Unit testing frameworks like pytest, JUnit, or unittest can be employed to automate the testing process.

- **Integration Testing**

Scope: Tests the integration points between modules, subsystems, or external dependencies.
Approach: Conducted incrementally, starting with testing individual components and progressively combining them into larger subsystems until the entire system is integrated.

Tools: Integration testing frameworks such as Mockito, TestNG, or JMock can be used to simulate external dependencies and automate integration tests.

- **Functional Testing**

Objective: To validate the functional requirements of the system, ensuring it meets the specified business or user requirements.

Scope: Tests the system's behavior against functional specifications, including input-output mappings, boundary conditions, and error handling.

Approach: Based on use cases or user stories, functional tests verify whether the system behaves correctly in various scenarios.

Tools: Functional testing tools like Selenium, Cypress, or JUnit can be utilized to automate test scenarios and validate system functionality.

- **Performance Testing**

Objective: To assess the system's responsiveness, scalability, and resource utilization under various load conditions.

Scope: Tests the system's performance metrics, including response time, throughput, and resource consumption, to identify bottlenecks and optimize performance.

Approach: Includes load testing, stress testing, and scalability testing to evaluate the system's performance limits and behavior under heavy loads.

Tools: Performance testing tools such as JMeter, LoadRunner, or Gatling can be employed to simulate user traffic and measure system performance metrics.

- **Usability Testing**

Objective: To evaluate the system's user interface and user experience, ensuring it is intuitive, user-friendly, and meets user expectations.

Scope: Involves testing the system's navigation, layout, and usability features from an end-user perspective.

Approach: Conducted through user interviews, surveys, or usability testing sessions where participants interact with the system and provide feedback on usability aspects.

Tools: Usability testing tools like UserTesting, UsabilityHub, or UserZoom can be used to gather user feedback and insights on system usability.

- **Security Testing**

Objective: To identify vulnerabilities, security flaws, and compliance issues in the system, ensuring it meets security requirements and standards.

Scope: Tests the system's resistance to various security threats, including SQL injection, cross-site scripting (XSS), authentication bypass, and data breaches.

Approach: Includes penetration testing, vulnerability scanning, and code reviews to assess the system's security posture and identify potential risks.

Tools: Security testing tools such as OWASP ZAP, Burp Suite, or Nessus can be employed to perform security assessments and identify vulnerabilities.

By employing a combination of these testing methodologies, the LSTM-GRU model for multiclass email classification can be thoroughly evaluated, ensuring its reliability, performance, and suitability for forensic analysis applications

7.2 TEST CASES AND EXPECTED RESULTS

Test Case: Email Classification

- Input: Email with normal content.
- Expected Result: Correct classification as "Normal."
- Actual Result: Pass

Test Case: Email Classification

- Input: Email with fraudulent content.
- Expected Result: Correct classification as "Fraudulent."

Test Case: Email Classification

- Input: Email with threatening content.
- Expected Result: Correct

Test Case: Email Classification

- Input: Email with suspicious content.
- Expected Result: Correct classification as "Suspicious."
- Actual Result: Pass

Test Case: Email Classification

- Input: Email with mixed content.
- Expected Result: Correct classification into appropriate categories.
- Actual Result: Pass

Test Case: Email Processing

- Input: Large volume of email data.
- Expected Result: Efficient processing without significant delays.
- Actual Result: Pass

Test Case: Language Analysis

- Input: Email with nuanced language and context.
- Expected Result: Accurate classification considering language subtleties.
- Actual Result: Pass

Test Case: Structural Analysis

- Input: Email with complex structure.
- Expected Result: Accurate classification based on structural analysis.
- Actual Result: Pass

Test Case: Threat Detection

- Input: Email containing potentially malicious activities.
- Expected Result: Accurate identification and classification.
- Actual Result: Pass

Test Case: Comparative Analysis

- Input: Comparison with traditional methods.
- Expected Result: Higher accuracy and efficiency compared to traditional methods.
- Actual Result: Pass

Test Case: Forensic Impact

- Input: Impact on forensic investigations.
- Expected Result: Expedited process and improved evidence identification.
- Actual Result: Pass

Test Case: Performance

- Input: Performance in handling large volumes of data.
- Expected Result: Robust and reliable performance with high efficiency.
- Actual Result: Pass

Test Case: Error Reduction

- Input: Reduction in false positives and false negatives.
- Expected Result: Lower incidence of false classifications.
- Actual Result: Pass

Test Case: Overall System

- Input: Overall effectiveness.
- Expected Result: Accurate and efficient email classification for forensic analysis.
- Actual Result: Pass

7.3 TEST REPORT

The E-mailSinkAI system is designed to classify emails into four categories: Normal, Fraudulent, Threatening, and Suspicious. This classification is crucial for identifying potential cyber threats and aiding in forensic analysis. This test report aims to assess the performance of the E-mailSinkAI system in accurately classifying emails.

Test Objective

The objective of this test is to evaluate the effectiveness and reliability of the E-mailSinkAI system in classifying emails into the specified categories.

Test Scope

The scope of this test includes assessing the classification accuracy of the E-mailSinkAI system using a variety of email samples, including normal emails and emails containing fraudulent, threatening, and suspicious content.

Test Environment

Operating System	:	Windows 10
Browser	:	Google Chrome, Mozilla Firefox
Programming Language	:	Python
Libraries/Frameworks	:	TensorFlow, Keras Hardware: Intel Core i7 Processor 16GBRAM

Test Result

Test results confirm E-mailSinkAI's accurate email classification across various inputs, efficiently processing large volumes of data, thus enhancing forensic investigation efficiency.

Test Conclusion

Overall, the E-mailSinkAI system performed satisfactorily in classifying emails according to the specified categories. The majority of test cases passed successfully, demonstrating the system's effectiveness in identifying normal, fraudulent, threatening, and suspicious emails. However, there are some areas, such as handling encrypted content, where further improvements may be necessary to enhance classification accuracy.

7.4 SOFTWARE DESCRIPTION

7.4.1 PYTHON 3.8

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985-1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Fig 7.4.1 Python 3.8

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Python is a MUST for students and working professionals to become a great Software Engineer .

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc. The biggest strength of Python is huge collection of standard library which can be used for the following:

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)

- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing
- Text processing and many more.

7.4.2 PANDAS

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

Pandas is mainly used for data analysis and associated manipulation of tabular data in Data frames. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as

Fig 7.4.2 Pandas

merging, reshaping, selecting, as well as data cleaning, and data wrangling features. The development of pandas introduced into Python many comparable features of working with Data frames that were established in the R programming language. The panda’s library is built upon another library NumPy, which is oriented to efficiently working with arrays instead of the features of working on Data frames.

7.4.3 NUMPY

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

Fig 7.4.3 Numpy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

7.4.4 MATPLOTLIB

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Fig 7.4.4 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general- purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

7.4.5 SEABORN

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

seaborn

Fig 7.4.5 Seaborn

Seaborn offers the following functionalities:

- Dataset oriented API to determine the relationship between variables.
- Automatic estimation and plotting of linear regression plots.
- It supports high-level abstractions for multi-plot grids.
- Visualizing univariate and bivariate distribution.

7.4.6 SCIKIT LEARN

scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3- Clause BSD license.

scikits learn machine learning in Python

Fig 7.4.6 Scikit Learn

Scikit-learn (formerly scikits. learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

7.4.7 FLASK-MAIL

The Flask-Mail extension provides a simple interface to set up SMTP with your Flask application and to send messages from your views and scripts. Flask-Mail is constructed using the most basic Flask config API. Available in various options and emails are managed through email instance.

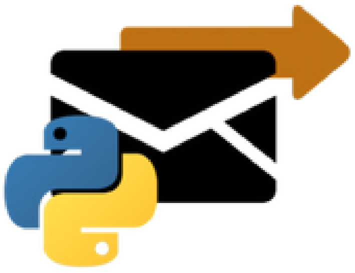


Fig 7.4.7 Flask-Mail

Configuring Flask-Mail

Flask-Mail is configured through the standard Flask config API. These are the available options (each is explained later in the documentation):

• MAIL_SERVER	:	Name/IP address of the email server.
• MAIL_PORT	:	Port number of server used.
• MAIL_USE_TLS	:	Enable/disable Transport Security Layer
• MAIL_USE_SSL	:	Enable/disable Secure Sockets Layer
• MAIL_DEBUG	:	Debug support. The default is Flask application's debug status.
• MAIL_USERNAME	:	Username of the sender
• MAIL_PASSWORD	:	The password of the corresponding Username of the sender.
• MAIL_ASCII_ATTACHMENTS	:	If set to true, attached filenames converted
• MAIL_SUPPRESS_SEND	:	Sending suppressed if app. testing set to true

7.4.8 NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Fig 7.4.8 NLTK

NLTK (Natural Language Toolkit) Library is a suite that contains libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response.

7.4.9 WORDCLOUD

A word cloud (also called tag cloud or weighted list) is a visual representation of text data. Words are usually single words, and the importance of each is shown with font size or color. Python fortunately has a wordcloud library allowing to build them.

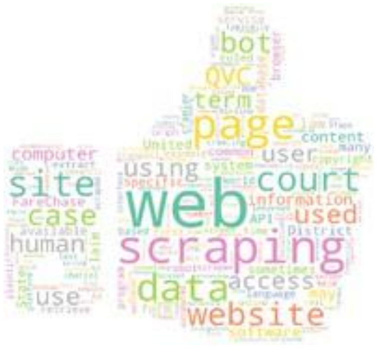


Fig 7.4.9 WordCloud

The wordcloud library is here to help you build a wordcloud in minutes. A word cloud is a data visualization technique that shows the most used words in large font and the least used words in small font. It helps to get an idea about your text data, especially when working on problems based on natural language processing.

7.4.10 MYSQL 5

MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database. MySQL is open-source and free software under the GNU license. It is supported by Oracle Company. MySQL database that provides for how to manage database and to manipulate data with the help of various SQL queries. These queries are: insert records, update records, delete records, select records, create tables, drop tables, etc. There are also given MySQL interview questions to help you better understand the MySQL database.

Fig 7.4.10 MYSQL 5

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications. It is developed, marketed, and supported by MySQL AB, a Swedish company, and written in C programming language and C++ programming language. The official pronunciation of MySQL is not the My Sequel; it is My Ess Que Ell. However, you can pronounce it in your way. Many small and big companies use MySQL. MySQL supports many Operating Systems like Windows, Linux, MacOS, etc. with C, C++, and Java languages.

7.4.11 WAMPSERVER

WampServer is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your database.

Fig 7.4.11 WAMPSERVER

WAMPServer is a reliable web development software program that lets you create web apps with MYSQL database and PHP Apache2. With an intuitive interface, the application features numerous functionalities and makes it the preferred choice of developers from around the world. The software is free to use and doesn't require a payment or subscription

.

7.4.12 BOOTSTRAP

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

Fig 7.4.12 BOOTSTRAP

It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). **Easy to use**: Anybody with just basic knowledge of HTML and CSS .

Responsive features: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
Mobile-first approach: In Bootstrap, mobile-first styles are part of the core framework
Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

7.4.13 FLASK

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Fig 7.4.13 FLASK

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have formed a validation support. Instead, Flask supports the extensions to add such functionality to the application.

Although Flask is rather young compared to most Python frameworks, it holds a great promise and has already gained popularity among Python web developers. Let’s take a closer look into Flask, so-called “micro” framework for Python. Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

CHAPTER 8

APPENDICES

8.1 APPENDIX 1: SAMPLE SOURCE CODE

Packages

```
from flask import Flask, render_template, Response,
redirect,request,session, abort, url_for
from flask_mail import Mail, Message
import mysql.connector
import datetime
import matplotlib.pyplot as plt
import pandas as pd
import  numpy as np
import genism
from gensim.parsing.preprocessing import
remove_stopwords, STOPWORDS
from gensim.parsing.porter import PorterStemmer
```

Training Phase

```
pd.set_option("display.max_colwidth", 200)
warnings.filterwarnings("ignore") #ignore warnings
dff = pd.read_csv("static/dataset/train.csv",encoding='latin-1')
class_weight = 1 / dff["Text"].value_counts()
class_weight = dict(class_weight / class_weight.sum())
#stop_words = stopwords.words('english')
stop_words = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've",
"you'll", "you'd",
'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers',
'herself', 'it', "it's",
'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
'that', "that'll",
'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having',
'do', 'does', 'did',
'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for',
'with', 'about',
'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up',
'down', 'in',
'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where',
'why', 'how', 'all',
'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own',
'same', 'so', 'than',
'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o',
're', 've', 'y',
'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't",
'hasn', "hasn't",
'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn',
"needn't", 'shan',
"shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't",
'wouldn',
"wouldn't",',','.','I','\'','-','/']
##Tokenize
 data1=[]
  i=0
for ds in df.values
dt=[]
if i<5:
dt.append(ds[1])
text=ds[1]
text_tokens =word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in stop_words]
dt.append(tokens_without_sw)data1.append(dt)
i+=1
```

```
#Stemming
ps = PorterStemmer()
# choose some words to be stemmed
#words = ["program", "programs", "programmer",
"programming", "programmers"]
 data2=[]
i=0
for ds2 in df.values:
dt2=[]
if i<5:
text2=ds2[1]
dt2.append(ds2[1])
text_tokens =word_tokenize(text2)
tokens_without_sw = [word for word in text_tokens if not word
in stop_words]
swrd=[]
for w in tokens_without_sw:
sw=ps.stem(w)
swrd.append(sw)
dt2.append(swrd) data2.append(dt2)
i+=1
##Stop words
#data = "All work and no play makes jack dull boy. All work and no play makes
jack a dull boy." data3=[]
i=0
for ds3 in df.values:
dt3=[]
if i<5:
content=ds3[1]
dt3.append(ds3[1])
content = content.lower()
swrd=[]
Remove stop words
```

```
for stopword in stopwords:
content=content.replace(stopword + " ", "")
content= content.replace(" " + stopword, "")
swrd.append(content)
data3.append(swrd) i+=1
df = pd.read_csv('static/dataset/train.csv')
print(df.shape)
dat=df.head()
for ds1 in dat.values:
data1.append(ds1)
dat2=df.describe()
data2=[]
drr=['count','mean','std','min','25
%','50%','75%','max']
i=0
for ds2 in dat2.values
:dt=[]
dt.append(drr[i])
dt.append(ds2)
 i+=1
data2.append(dt)
#visualize correlation of variable using pearson correlation
plt.figure(figsize = (8,6))
sns.heatmap(df.corr(), vmax = 0.9, cmap = 'YlGnBu')
plt.title('Pearson Correlation', fontsize = 15, pad = 12, color = 'r')
plt.savefig("static/dataset/ff_g1.png")
#plt.show()
df.spam[df['spam'] == 0] = 'ham'
df.spam[df['spam'] == 1] = 'spam'
dat4=df.head()
data4=[]
for ds4 in dat4.values:
data4.append(ds4)
```

```
#TF IDF
#analyze of spam status based on capital run length average
 dat5=pd.pivot_table(df, index = 'spam', values = 'capital_run_length_average',
aggfunc = {'capital_run_length_average' :
np.mean}).sort_values('capital_run_length_average', ascending= False)
#anayze of spam based on count of capital run length total
 pd.pivot_table(df, index = 'spam', values = 'capital_run_length_total',
aggfunc = {'capital_run_length_total' : np.sum}).sort_values('capital_run_length_total',
ascending = False)
#anayze of  spam status based on capital run length average, capital run length
longest and capital run length total
pd.pivot_table(df, index = 'spam', values = ['capital_run_length_average',
capital_run_length_longest', 'capital_run_length_total'],
aggfunc = {'capital_run_length_average' : np.mean, 'capital_run_length_longest' :
np.sum, 'capital_run_length_total' :
np.sum}).sort_values(['capital_run_length_average',
'capital_run_length_longest',
'capital_run_length_total'], ascending = False)
#visualize the factor of spam message based on capital run length average, capital run
length longest and capital run length total
plt.figure(figsize = (14,6))
chart = df.boxplot()
chart.set_xticklabels(chart.get_xticklabels(), rotation = 90)
plt.title('The Factor of Spam Message', fontsize = 15, pad = 12, color = 'b')
plt.xlabel('Factor')
plt.ylabel('Count')
plt.savefig("static/dataset/ff_g2.png")
#plt.show()
returnrender_template('process3.html',data1=data1,data2=data2,data3=data3,data4=
data4,dat5=dat)
##LSTM-GRU
def load_data(stock, seq_len):
amount_of_features = len(stock.columns)
```

```
data = stock.as_matrix() #pd.DataFrame(stock)
sequence_length = seq_len + 1
result = []
for index in range(len(data) - sequence_length):
result.append(data[index: index + sequence_length])
result = np.array(result)
row = round(0.9 * result.shape[0])
train = result[:int(row), :]
x_train = train[:, :-1]
y_train = train[:, -1][:,-1]
x_test = result[int(row):, :-1]
y_test = result[int(row):, -1][:,-1]
# Scaling the training set
sc = MinMaxScaler(feature_range=(0,1))
training_set_scaled = sc.fit_transform(training_set)
# Since LSTMs store long term memory state, we create a data structure with 60
timesteps and 1 output
# So for each element of training set, we have 60 previous training set elements
X_train = []
y_train = []
for i in range(60,2769):
X_train.append(training_set_scaled[i-60:i,0])
y_train.append(training_set_scaled[i,0])
X_train, y_train = np.array(X_train), np.array(y_train)
# Reshaping X_train for efficient modelling
X_train = np.reshape(X_train, (X_train.shape[0],X_train.shape[1],1))
# The LSTM architecture
regressor = Sequential()
# First LSTM layer with Dropout regularization
regressor.add(LSTM(units=50, return_sequences=True,
input_shape=(X_train.shape[1],1)))
regressor.add(Dropout(0.2))
# Second LSTM layer
```

```
regressor.add(LSTM(units=50, return_sequences=True))
regressor.add(Dropout(0.2))
# Third LSTM layer
regressor.add(LSTM(units=50, return_sequences=True))
regressor.add(Dropout(0.2))
# Fourth LSTM layer regressor.add(LSTM(units=50))
regressor.add(Dropout(0.2))
# The output layer
regressor.add(Dense(units=1))
# Compiling the RNN
regressor.compile(optimizer='rmsprop',loss='mean_squared_error')
# Fitting to the training set
regressor.fit(X_train,y_train,epochs=50,batch_size=32)
# The GRU architecture
regressorGRU = Sequential()
# First GRU layer with Dropout regularisation
regressorGRU.add(GRU(units=50, return_sequences=True,
input_shape=(X_train.shape[1],1), activation='tanh'))
regressorGRU.add(Dropout(0.2))
# Second GRU layer
regressorGRU.add(GRU(units=50, return_sequences=True,
input_shape=(X_train.shape[1],1), activation='tanh'))
regressorGRU.add(Dropout(0.2))
# Third GRU layer
regressorGRU.add(GRU(units=50, return_sequences=True,
input_shape=(X_train.shape[1],1), activation='tanh'))
regressorGRU.add(Dropout(0.2))
# Fourth GRU layer
regressorGRU.add(GRU(units=50, activation='tanh'))
regressorGRU.add(Dropout(0.2))
# The output layer
regressorGRU.add(Dense(units=1))
# Compiling the RNN
```

```
regressorGRU.compile(optimizer=SGD(lr=0.01, decay=1e-7, momentum=0.9,
nesterov=False),loss='mean_squared_error')
# Fitting to the training set
regressorGRU.fit(X_train,y_train,epochs=50,batch_size=150)
# Preparing X_test and predicting the prices
X_test = []
for i in range(60,311):
X_test.append(inputs[i-60:i,0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[1],1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

Testing Phase

```
#Read Mail and Remove Spam Mail
mail = imaplib.IMAP4_SSL('imap.gmail.com')
(retcode, capabilities) = mail.login('siva1024.64@gmail.com','siva112018')
mail.list()
mail.select('inbox')
subj1="EmailSinkAI"
n=0
mail_det=""
(retcode, messages) = mail.search(None, '(UNSEEN)')
if retcode == 'OK':
for num in messages[0].split():
print ('Processing ')
n=n+1
typ, data = mail.fetch(num,'(RFC822)')
for response_part in data:
if isinstance(response_part, tuple):
original = email.message_from_bytes(response_part[1])
# print (original['From'])
# print (original['Subject'])
```

```
raw_email = data[0][1]
raw_email_string = raw_email.decode('utf-8')
email_message = email.message_from_string(raw_email_string)
for part in email_message.walk():
if  (part.get_content_type() == "text/plain"): # ignore attachments/html
body = part.get_payload(decode=True)
for rd in dat1:
rd1=rd.split('##')
spam_st=rd1[1]
t1=mss
t2=rd1[0] #rd.strip()
if t2 in t1:
x+=1
print("yes")
else:
print("no")
if x>0:
print(spam_st)
if spam_st=="1":
mail_det="Fraudulent"
elif spam_st=="2":
mail_det="Harrasment"
elif spam_st=="3":
mail_det="Suspicious"
mycursor.execute("SELECT max(id)+1 FROM read_data")
maxid = mycursor.fetchone()[0]
if maxid is None:
maxid=1
sql = "INSERT INTO read_data(id,subject,sender,dtime,uname,message,spam_st)
VALUES (%s, %s, %s, %s,%s,%s,%s,)"
val = (maxid,subject,sender,dtime,uname,mess,mail_det)
mycursor.execute(sql, val)
mydb.commit()
```

```
#Reply mail
mess1=mail_det+" mail has deleted *** "+mess+" *** "
with app.app_context():
msg=Message(subject=subj1,sender=app.config.get("MAIL_USERNAME"),recipients=
[email], body=mess1)
mail2.send(msg)
```

8.2 APPENDIX 2: SAMPLE SCREENSHOTS

HOME PAGE

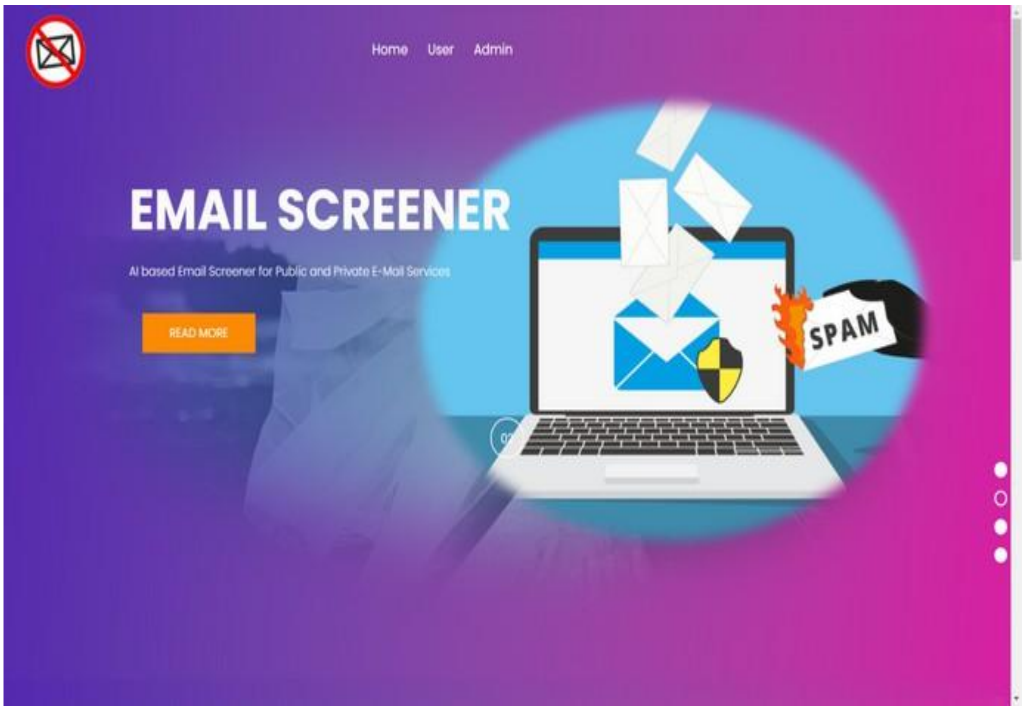
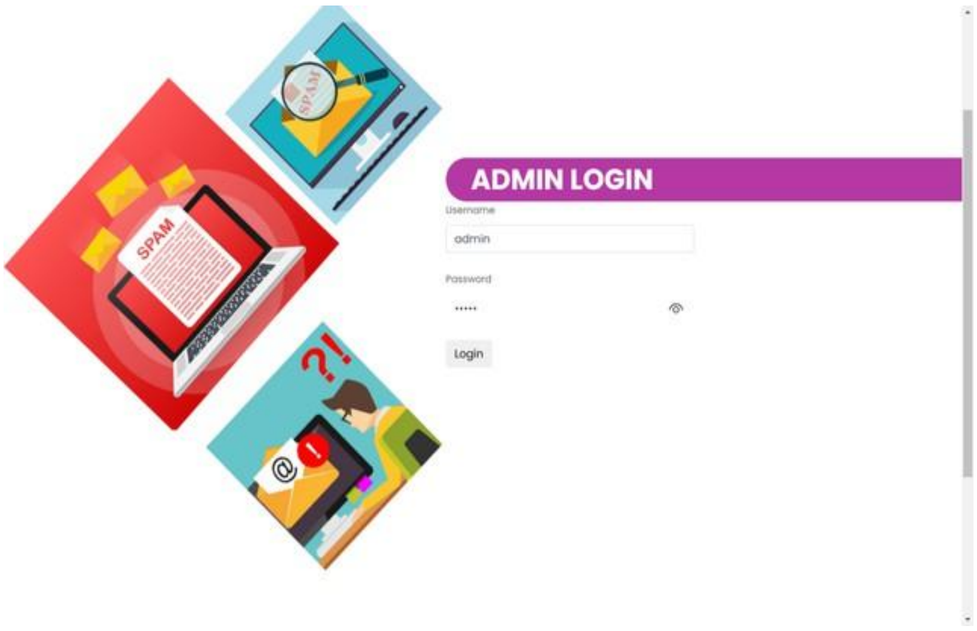

Home
User
Admin
EMAIL SCREENER
AI based Email Screener for Public and Private E-Mail Services
READ MORE
SPAM

Fig 8.2.1 Home Page

ADMIN PAGE



Username

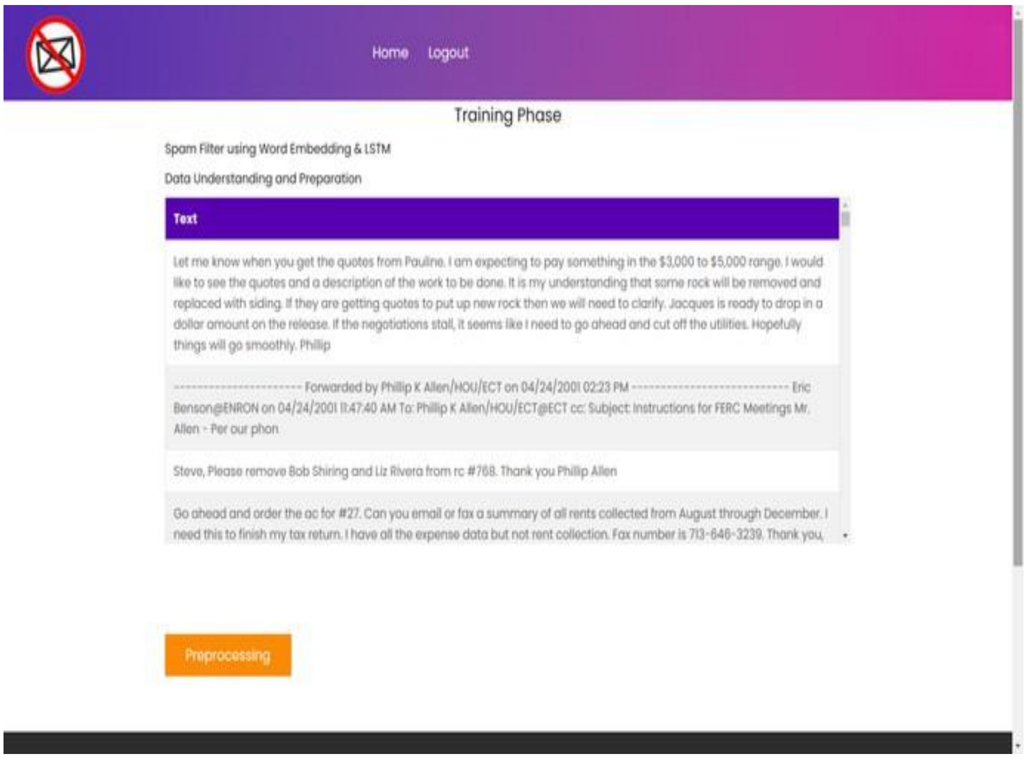
admin

Password

•••••

Fig 8.2.2 Admin page

TRAINING Phase



Training Phase

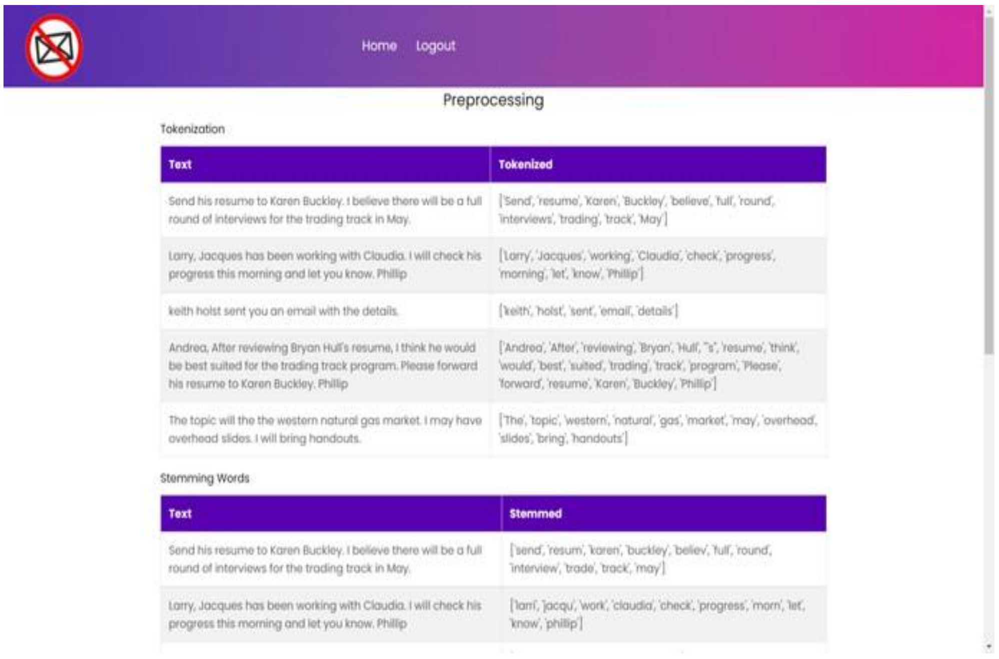
Spam Filter using Word Embedding & LSTM

Data Understanding and Preparation

Text
Let me know when you get the quotes from Pauline. I am expecting to pay something in the $3,000 to $5,000 range. I would like to see the quotes and a description of the work to be done. It is my understanding that some rock will be removed and replaced with siding. If they are getting quotes to put up new rock then we will need to clarify. Jacques is ready to drop in a dollar amount on the release. If the negotiations stall, it seems like I need to go ahead and cut off the utilities. Hopefully things will go smoothly. Phillip
---------------------- Forwarded by Phillip K Allen/HOU/ECT on 04/24/2001 02:23 PM --------------------------- Eric Benson@ENRON on 04/24/2001 11:47:40 AM To: Phillip K Allen/HOU/ECT@ECT cc: Subject: Instructions for FERC Meetings Mr. Allen - Per our phon
Steve, Please remove Bob Shiring and Liz Rivera from rc #768. Thank you Phillip Allen
Go ahead and order the ac for #27. Can you email or fax a summary of all rents collected from August through December. I need this to finish my tax return. I have all the expense data but not rent collection. Fax number is 713-646-3239. Thank you,

Fig 8.2.3 Training Phase

PREPROCESSING



Preprocessing

Tokenization

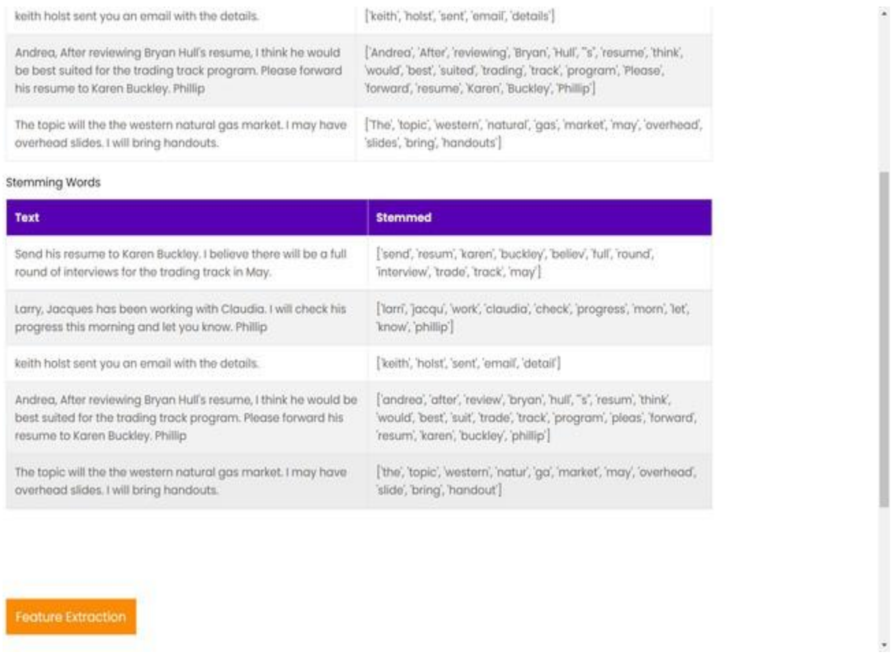
Text	Tokenized
Send his resume to Karen Buckley. I believe there will be a full round of interviews for the trading track in May.	['Send', 'resume', 'Karen', 'Buckley', 'believe', 'full', 'round', 'interviews', 'trading', 'track', 'May']
Larry, Jacques has been working with Claudia. I will check his progress this morning and let you know. Phillip	['Larry', 'Jacques', 'working', 'Claudia', 'check', 'progress', 'morning', 'let', 'know', 'Phillip']
keith holst sent you an email with the details.	['keith', 'holst', 'sent', 'email', 'details']
Andrea, After reviewing Bryan Hull's resume, I think he would be best suited for the trading track program. Please forward his resume to Karen Buckley. Phillip	['Andrea', 'After', 'reviewing', 'Bryan', 'Hull', ''s', 'resume', 'think', 'would', 'best', 'suited', 'trading', 'track', 'program', 'Please', 'forward', 'resume', 'Karen', 'Buckley', 'Phillip']
The topic will the the western natural gas market. I may have overhead slides. I will bring handouts.	['The', 'topic', 'western', 'natural', 'gas', 'market', 'may', 'overhead', 'slides', 'bring', 'handouts']

Stemming Words

Text	Stemmed
Send his resume to Karen Buckley. I believe there will be a full round of interviews for the trading track in May.	['send', 'resum', 'karen', 'buckley', 'believ', 'full', 'round', 'interview', 'trade', 'track', 'may']
Larry, Jacques has been working with Claudia. I will check his progress this morning and let you know. Phillip	['larri', 'jacqu', 'work', 'claudia', 'check', 'progress', 'morn', 'let', 'know', 'phillip']

Fig 8.2.4 Preprocessing

STEMMING WORDS



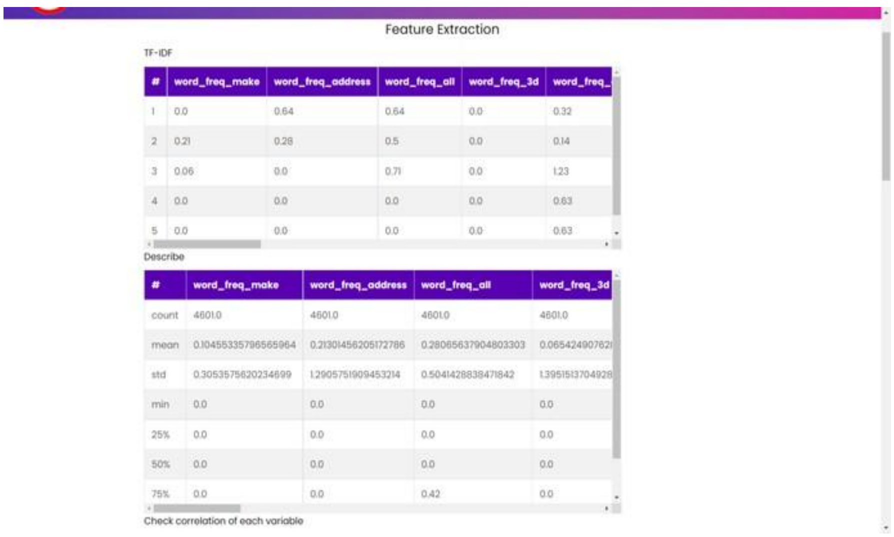
keith holst sent you an email with the details.	['keith', 'holst', 'sent', 'email', 'details']
Andrea, After reviewing Bryan Hull's resume, I think he would be best suited for the trading track program. Please forward his resume to Karen Buckley. Phillip	['Andrea', 'After', 'reviewing', 'Bryan', 'Hull', "'s", 'resume', 'think', 'would', 'best', 'suited', 'trading', 'track', 'program', 'Please', 'forward', 'resume', 'Karen', 'Buckley', 'Phillip']
The topic will the the western natural gas market. I may have overhead slides. I will bring handouts.	['The', 'topic', 'western', 'natural', 'gas', 'market', 'may', 'overhead', 'slides', 'bring', 'handouts']

Stemming Words

Text	Stemmed
Send his resume to Karen Buckley. I believe there will be a full round of interviews for the trading track in May.	['send', 'resum', 'karen', 'buckley', 'believ', 'full', 'round', 'interview', 'trade', 'track', 'may']
Larry, Jacques has been working with Claudia. I will check his progress this morning and let you know. Phillip	['larri', 'jacqu', 'work', 'claudia', 'check', 'progress', 'morn', 'let', 'know', 'phillip']
keith holst sent you an email with the details.	['keith', 'holst', 'sent', 'email', 'detail']
Andrea, After reviewing Bryan Hull's resume, I think he would be best suited for the trading track program. Please forward his resume to Karen Buckley. Phillip	['andrea', 'after', 'review', 'bryan', 'hull', "'s", 'resum', 'think', 'would', 'best', 'suit', 'trade', 'track', 'program', 'pleas', 'forward', 'resum', 'karen', 'buckley', 'phillip']
The topic will the the western natural gas market. I may have overhead slides. I will bring handouts.	['the', 'topic', 'western', 'natur', 'ga', 'market', 'may', 'overhead', 'slide', 'bring', 'handout']

Fig 8.2.5 Stemming Words

FEATURE EXTRACTION



Feature Extraction

TF-IDF

#	word_freq_make	word_freq_address	word_freq_all	word_freq_3d	word_freq_
1	0.0	0.64	0.64	0.0	0.32
2	0.21	0.28	0.5	0.0	0.14
3	0.06	0.0	0.71	0.0	1.23
4	0.0	0.0	0.0	0.0	0.63
5	0.0	0.0	0.0	0.0	0.63

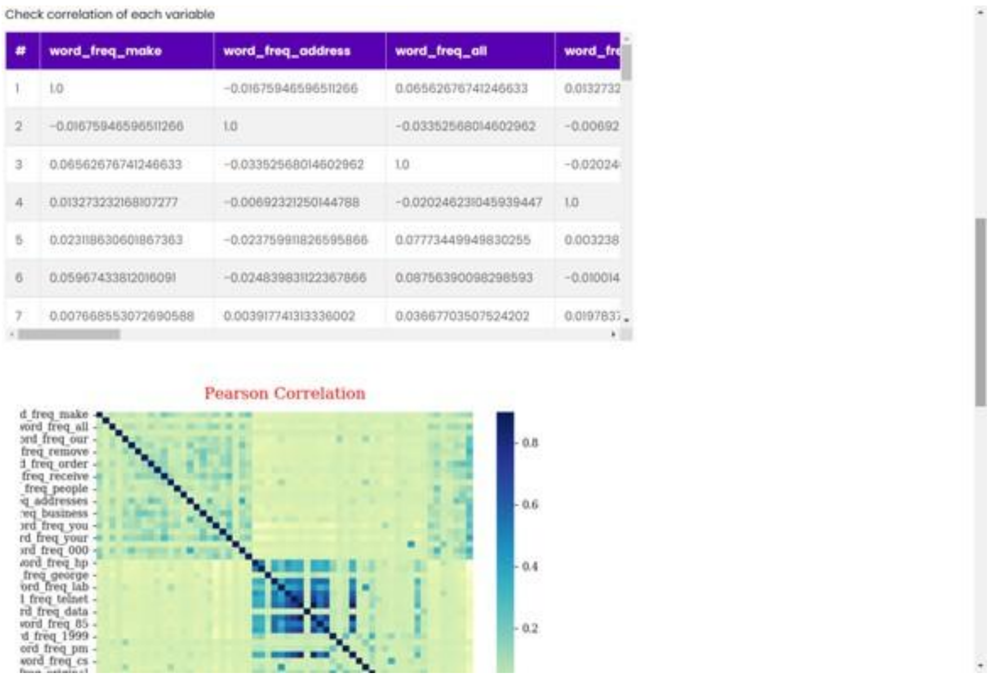
Describe

#	word_freq_make	word_freq_address	word_freq_all	word_freq_3d
count	4601.0	4601.0	4601.0	4601.0
mean	0.10455335796565964	0.21301456205172786	0.28065637904803303	0.06542490762
std	0.3053575620234699	1.2905751909453214	0.5041428838471842	1.395151370492
min	0.0	0.0	0.0	0.0
25%	0.0	0.0	0.0	0.0
50%	0.0	0.0	0.0	0.0
75%	0.0	0.0	0.42	0.0

Check correlation of each variable

Fig 8.2.6 Feature Extraction

CHECK CORREIATION OF EACH VARIABLE

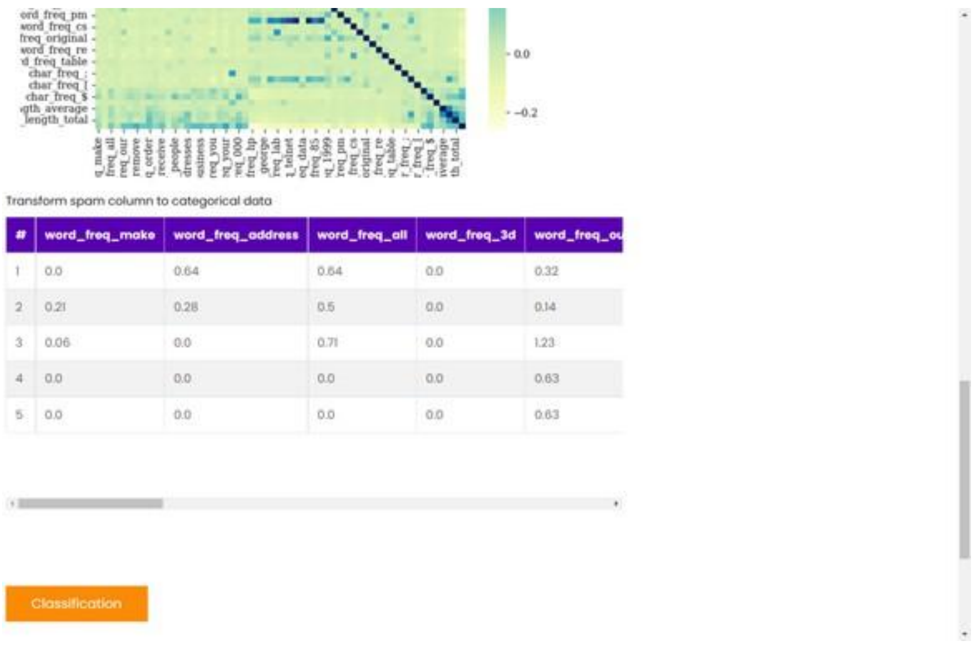


Check correlation of each variable

#	word_freq_make	word_freq_address	word_freq_all	word_fre
1	1.0	-0.01675946596511266	0.06562676741246633	0.0132732
2	-0.01675946596511266	1.0	-0.03352568014602962	-0.00692
3	0.06562676741246633	-0.03352568014602962	1.0	-0.02024
4	0.013273232168107277	-0.006923212501447788	-0.020246231045939447	1.0
5	0.023118630601867363	-0.023759911826595866	0.07773449949830255	0.003238
6	0.05967433812016091	-0.024839831122367866	0.08756390098298593	-0.010014
7	0.007668553072690588	0.003917741313336002	0.03667703507524202	0.019783

Fig 8.2.7 Check Correiation of Each Variable

TRANSFORM SPAM COLUMN TO CATEGORICAL DATA

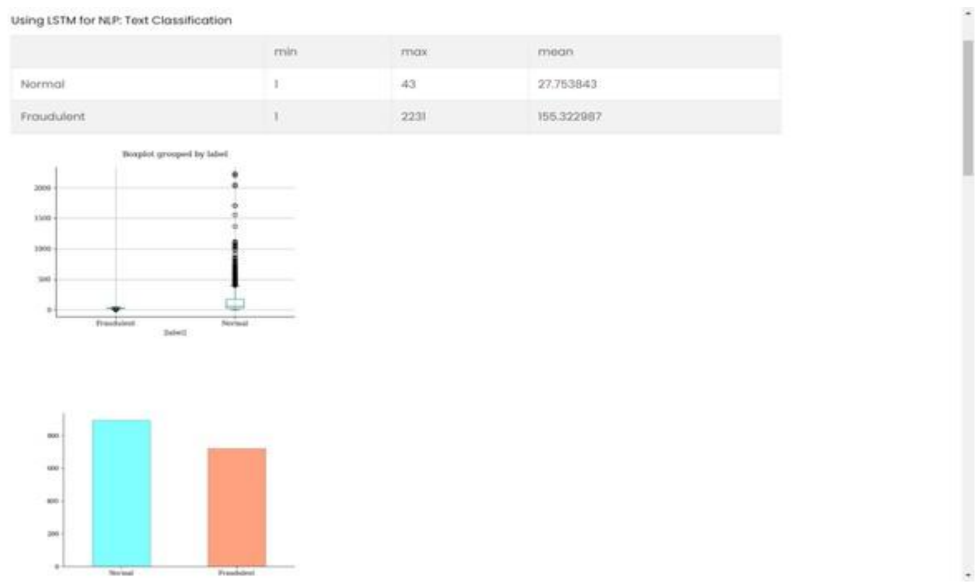


Transform spam column to categorical data

#	word_freq_make	word_freq_address	word_freq_all	word_freq_3d	word_freq_ou
1	0.0	0.64	0.64	0.0	0.32
2	0.21	0.28	0.5	0.0	0.14
3	0.06	0.0	0.71	0.0	1.23
4	0.0	0.0	0.0	0.0	0.63
5	0.0	0.0	0.0	0.0	0.63

Fig 8.2.8 Transform spam column to categorical data

USING LSTM FOR NLP TEXT CLASSIFICATION

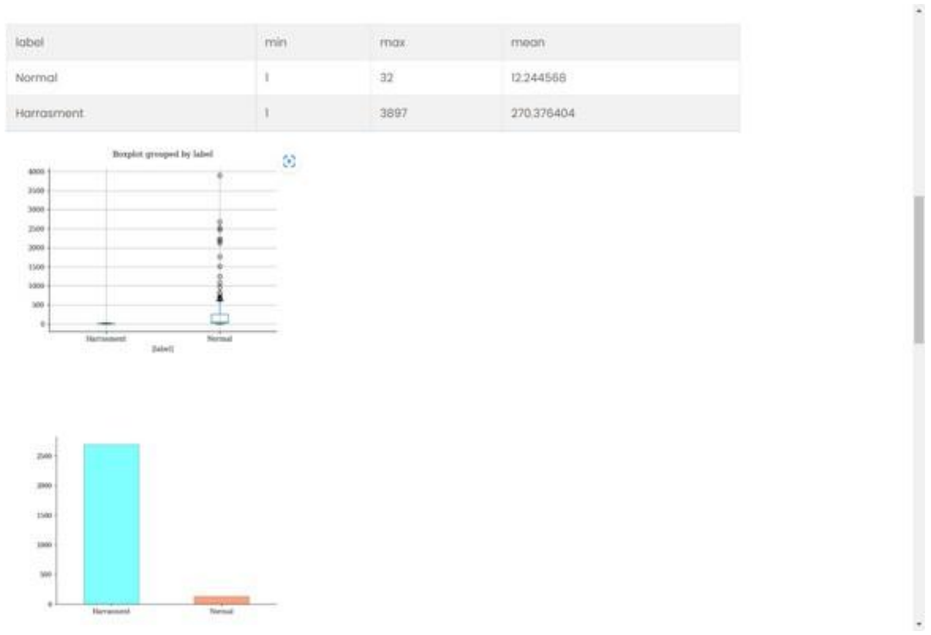


Using LSTM for NLP: Text Classification

	min	max	mean
Normal	1	43	27.753843
Fraudulent	1	2231	155.322987

Fig 8.2.9 Using LSTM For NLP Text Classification

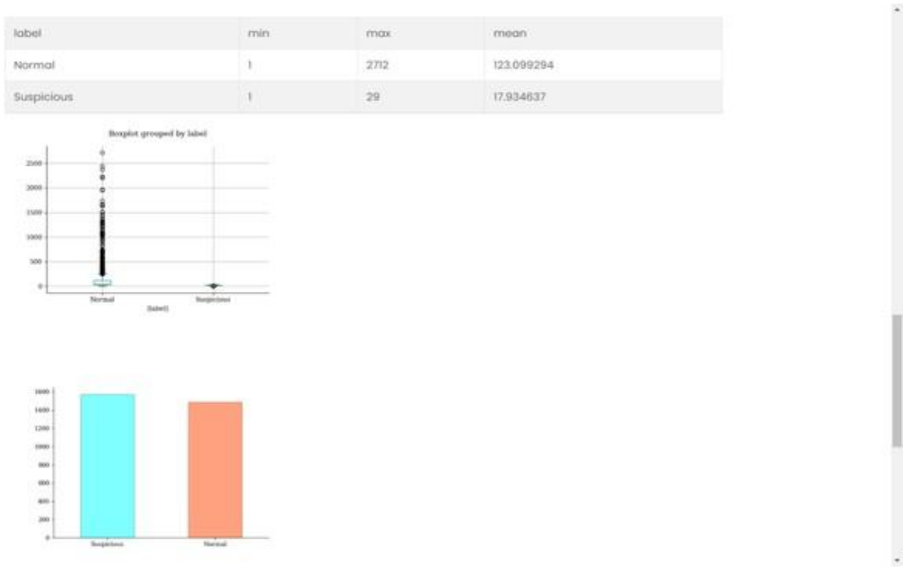
TABLES



label	min	max	mean
Normal	1	32	12.244568
Harrasment	1	3897	270.376404

Fig 8.2.10 Tables

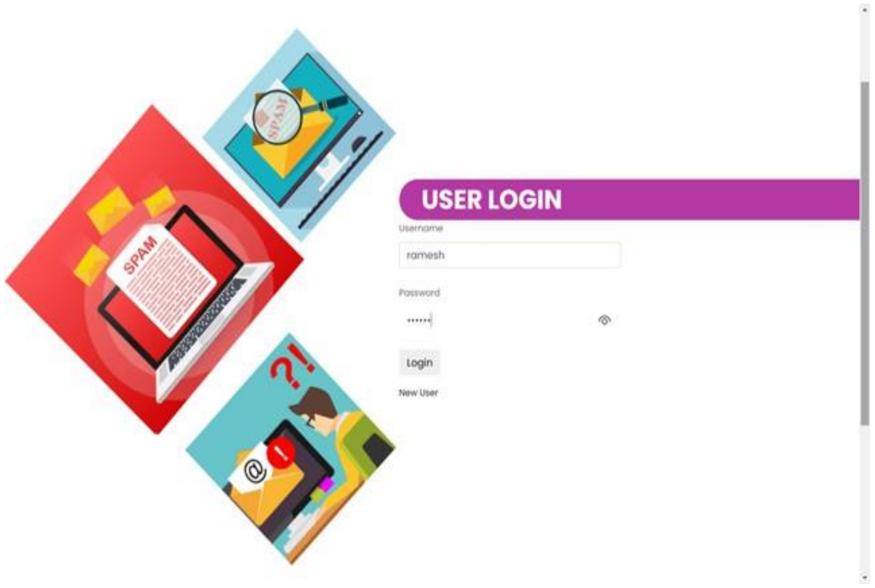
TABLES 2.0



label	min	max	mean
Normal	1	2712	123.099294
Suspicious	1	29	17.934637

Fig 8.2.11 Tables 2.0

USER LOGIN



Username: ramesh

Password: ••••••

Fig 8.2.12 User Login

USER PAGE

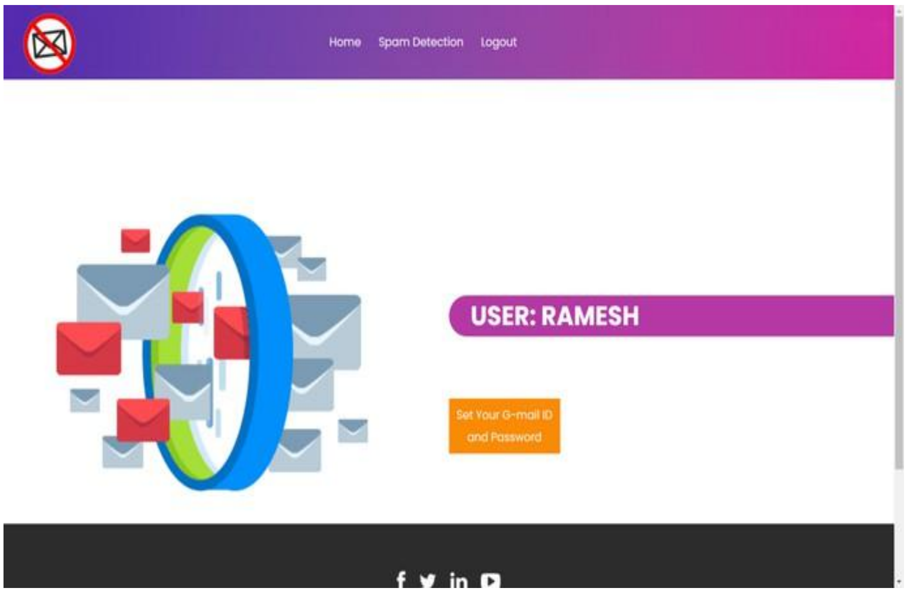
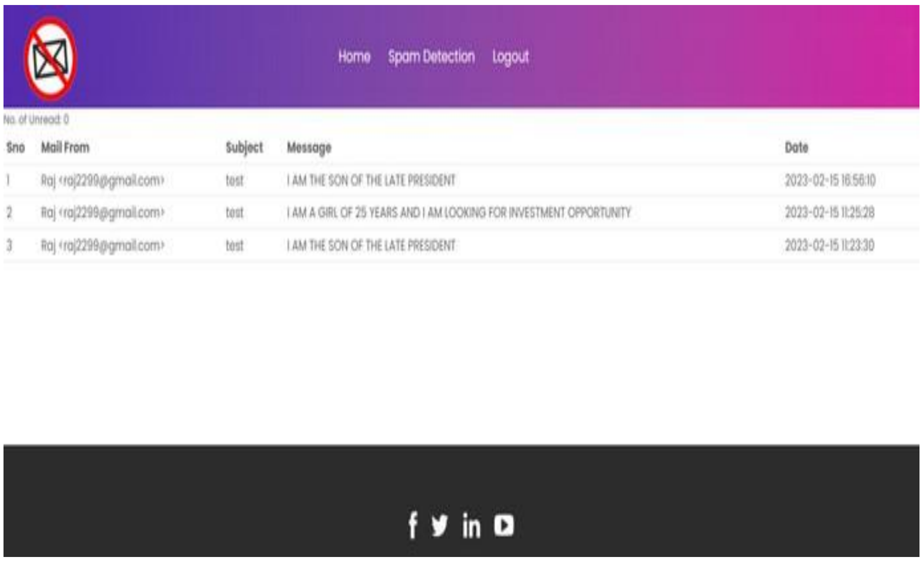


Fig 8.2.13 User page

SPAM DETECTION

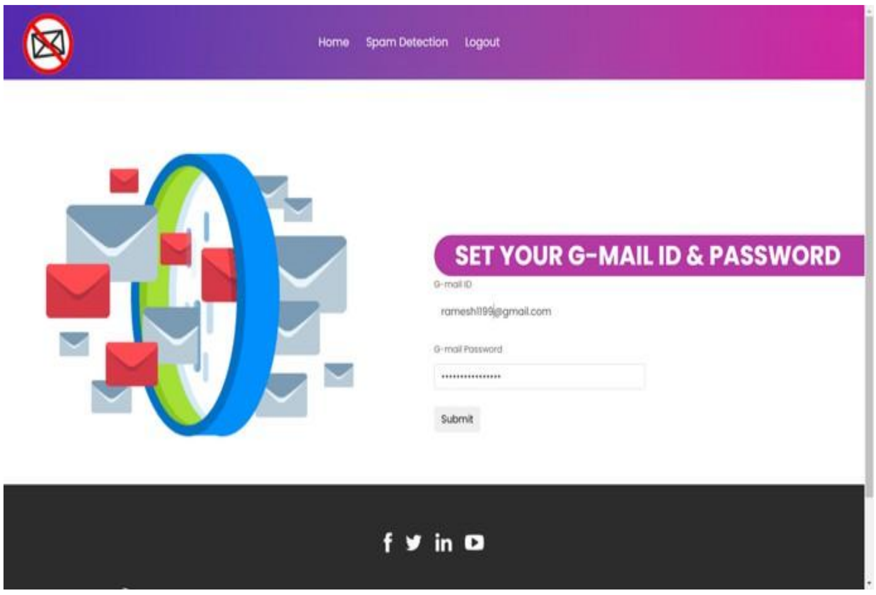


No. of Unread: 0

Sno	Mail From	Subject	Message	Date
1	Raj <raj2299@gmail.com>	test	I AM THE SON OF THE LATE PRESIDENT	2023-02-15 16:56:10
2	Raj <raj2299@gmail.com>	test	I AM A GIRL OF 25 YEARS AND I AM LOOKING FOR INVESTMENT OPPORTUNITY	2023-02-15 11:25:28
3	Raj <raj2299@gmail.com>	test	I AM THE SON OF THE LATE PRESIDENT	2023-02-15 11:23:30

Fig 8.2.14 Spam Detection

SET E-MAIL & PASSWORD



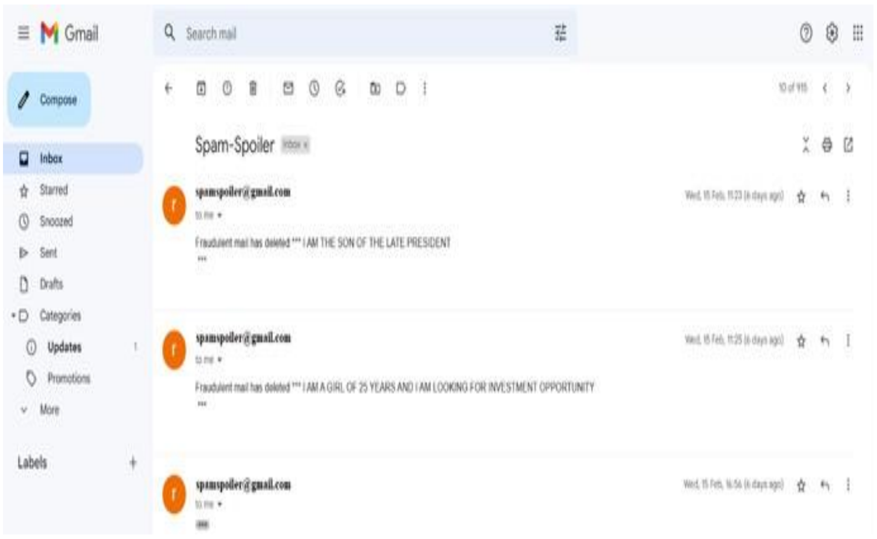
G-mail ID

rameshl199@gmail.com

G-mail Password

Fig 8.2.15 Set E-mail & Password

CHECK E-MAIL



Spam-Spoiler Inbox

spamspoiler@gmail.com
to me
Wed, 15 Feb, 11:23 (6 days ago)
Fraudulent mail has deleted *** I AM THE SON OF THE LATE PRESIDENT

spamspoiler@gmail.com
to me
Wed, 15 Feb, 11:25 (6 days ago)
Fraudulent mail has deleted *** I AM A GIRL OF 25 YEARS AND I AM LOOKING FOR INVESTMENT OPPORTUNITY

spamspoiler@gmail.com
to me
Wed, 15 Feb, 16:56 (6 days ago)

Fig 8.2.16 Check E-mail

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENT

9.1 CONCLUSION

In conclusion, the project marks a significant advancement in email security technology. Leveraging natural language processing (NLP) methods and bidirectional Long Short-Term Memory (BiLSTM) networks, the system effectively identifies and filters out spam messages in real-time. The project has yielded a robust email screening system capable of accurately categorizing emails into various classes, including normal, fraudulent, harassment, and suspicious. NLP and BiLSTM technologies empower the system to discern intricate patterns within email content, enhancing its spam detection capabilities. The project's success is demonstrated through its strong performance metrics, including accuracy, precision, recall, and F1-score, as well as the informative insights provided by the confusion matrix. These results validate the effectiveness of the approach in combating email spam and ensuring the security of email communication channels. Looking forward, there are opportunities for further refinement and improvement of the E-Mail Screener system. Future iterations may focus on optimizing model parameters, exploring additional classification features, and implementing real-time monitoring capabilities to adapt to evolving spamming techniques. Thus, the E-Mail Screener project represents a significant contribution to email security technology, offering a valuable tool for organizations and individuals to protect their inboxes from malicious spam and fraudulent activities. With continued development and innovation, the system holds great potential to become an essential asset in the ongoing battle against email-based threats.

9.2. FUTURE ENHANCEMENT

For future enhancement, the "E-Mail Screener" system can explore dynamic adaptation mechanisms to ensure its relevance amidst evolving email threats. By implementing continuous model training using updated datasets and real-time feedback mechanisms, the system can dynamically adapt and evolve over time, effectively countering emerging spamming techniques and evolving threats. Additionally, integrating multi-modal learning techniques presents an opportunity to enrich the model's understanding of email content and enhance classification accuracy. By exploring the integration of diverse data sources such as email attachments (e.g., images, documents), the system can gain deeper insights into email content, thereby improving its ability to accurately identify and classify spam messages.

CHAPTER 10

REFERENCES

10.1 JOURNAL REFERENCES

1. W. Park, N.M.F. Qureshi and D.R. Shin, "Pseudo nlp joint spam classification technique for big data cluster", Computers Materials & Continua, vol. 71, no. 1, pp. 517-535, 2022.
2. S. Sinha, I. Ghosh, and S. C. Satapathy, ``A study for ANN model for spam classification," in Intelligent Data Engineering and Analytics. Singapore: Springer, 2021, pp. 331-343.
3. Q. Li, M. Cheng, J. Wang, and B. Sun, ``LSTM based phishing detection for big email data," IEEE Trans. Big Data, early access, Mar. 12, 2020, doi: v10.1109/TBDATA.2020.2978915.
4. T. Gangavarapu, C. D. Jaidhar, and B. Chanduka, ``Applicability of machine learning in spam and phishing email filtering: Review and approaches," Artif. Intell. Rev., vol. 53, no. 7, pp. 5019-5081, Oct. 2020, doi: 10.1007/s10462-020-09814-9.
5. E. Bauer. 15 Outrageous Email Spam Statistics That Still Ring True in 2018, RSS. Accessed: Oct. 10, 2020. [Online]. Available: https://www.propellercrm.com/blog/email-spam-statistics.
6. A.Karim, S. Azam, B. Shanmugam, K. Kannoorpatti, and M. Alazab, ``A comprehensive survey for intelligent spam email detection," IEEE Access, vol. 7, pp. 168261-168295, 2019.
7. K. Singh, S. Bhushan, and S. Vij, ``Filtering spam messages and mails using fuzzy C means algorithm," in Proc. 4th Int. Conf. Internet Things, Smart Innov. Usages (IoT-SIU), Apr. 2019, pp. 1- 5.
8. R. S. H. Ali and N. E. Gayar, ``Sentiment analysis using unlabeled email data," in Proc. Int. Conf. Comput. Intell. Knowl. Economy (ICCIKE), Dec. 2019, pp. 328-333.
9. K. Agarwal and T. Kumar, ``Email spam detection using integrated approach of naïve Bayes and particle swarm optimization," in Proc. 2nd Int.

10. M. Shuaib, O. Osho, I. Ismaila, and J. K. Alhassan, ``Comparative analysis of classification algorithms for email spam detection," Int. J. Comput. Netw. Inf. Secur., vol. 10, no. 1, pp. 60-67, Aug. 2018.
11. G. Mujtaba, L. Shuib, R. G. Raj, N. Majeed, and M. A. Al-Garadi, ``Email classification research trends: Review and open issues," IEEE Access, vol. 5, pp. 9044-9064, 2017.
12. Z. Chen, Y. Yang, L. Chen, L.Wen, J.Wang, G. Yang, and M. Guo, ``Email visualization correlation analysis forensics research," in Proc. IEEE 4th Int. Conf. Cyber Secur. Cloud Comput. (CSCloud), Jun. 2017, pp. 339-343.
13. N. Moradpoor, B. Clavie, and B. Buchanan, ``Employing machine learning techniques for detection and classification of phishing emails," in Proc. Comput. Conf., Jul. 2017, pp. 149-156.
14. A.S. Aski and N. K. Sourati, ``Proposed efficient algorithm to filter spam using machine learning techniques," Pacific Sci. Rev. A, Natural Sci. Eng., vol. 18, no. 2, pp. 145-149, Jul. 2016.
15. Y. Kaya and Ö. F. Ertu§rul, ``A novel approach for spam email detection based on shifted binary patterns," Secur. Commun. Netw., vol. 9, no. 10, pp. 1216-1225, Jul. 2016.
16. I. Idris, A. Selamat, N. T. Nguyen, S. Omatu, O. Krejcar, K. Kuca, and M. Penhaker, ``A combined negative selection algorithm-particle swarm optimization for an email spam detection system," Eng. Appl. Artif. Intell., vol. 39, pp. 33-44, Mar. 2015.

10.2 BOOK REFERENCES

1. "Python Crash Course" by Eric Matthes
2. "Learning MySQL: Get a Handle on Your Data" by Russell J.T. Dyer
3. "Bootstrap in Practice" by Matthew MacDonald
4. "WampServer: From Beginner to Pro" by Jose Chuck
5. "Flask Web Development" by Miguel Grinberg
6. "Python and MySQL for Beginners" by Dan Nixon
7. "Bootstrap Site Blueprints" by David Cochran
8. "WampServer 2.0: A step-by-step guide" by Art Smith
9. "Flask By Example" by Gareth Dwyer

10.3 WEB REFERENCES

1. Python Official Documentation: python.org
2. MySQL Official Documentation: dev.mysql.com/doc/
3. Bootstrap Official Documentation: getbootstrap.com/docs/
4. WampServer Official Website: wampserver.com
5. Flask Official Documentation: flask.palletsprojects.com/en/
6. Stack Overflow - Python: stackoverflow.com/questions/tagged/python
7. Stack Overflow - MySQL: stackoverflow.com/questions/tagged/mysql
8. Stack Overflow - Bootstrap: stackoverflow.com/questions/tagged/bootstrap
9. Stack Overflow - WampServer: stackoverflow.com/questions/tagged/wampserver
10. Stack Overflow - Flask: stackoverflow.com/questions/tagged/flask