2024 IEEE International Conference on Computing, Power and Communication Technologies (IC2PCT) | 979-8-3503-8354-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/IC2PCT60090.2024.10486769

Detecting Email Spam with NLP: A Machine Learning Approach

Pallavi Jain
SCSE, Galgotias University,
Greater Noida, India
pallavi.jain@galgotiasuniversity.edu.in

Shivang Singh
SCSE, Galgotias University,
Greater Noida, India
shivang.20scse1010436@galgotias university.edu.in

Chaitanya Kumar Saxena
SCSE, Galgotias University
Greater Noida, India
saxena.akarsh5@gmail.com

Abstract—The rapid increase of digital communication has led to an exponential increase in email traffic, with a significant part being unwanted spam messages. This project aims to combat this issue by employing machine learning methods, explicitly Natural Language Processing (NLP), to develop an efficient email spam detection system. The core of our approach involves the transformation of email messages into numerical vectors using NLP techniques. This vectorization process captures the semantic meaning of the messages, enabling us to analyze and classify them effectively. To achieve this, we utilize advanced text preprocessing methods, including tokenization, stop word removal, and feature extraction. The classification phase of our project leverages the Naive Bayes classifier, a proven algorithm for text classification tasks. Through training the model with a varied and labeled data set of email messages, we enable it to distinguish between legitimate messages and spam with high accuracy. This approach is computationally efficient, making it suitable for real- time email filtering. This project results in a robust email spam detection system capable of automatically flagging and filtering out unwanted messages, thus improving the user experience and reducing the risk associated with malicious email content. Our evaluation of the system's performance demonstrates its effectiveness in achieving a high detection accuracy rate while minimizing false positives, ensuring that legitimate emails are not erroneously categorized as spam. The project underscores the potential of NLP and machine learning in enhancing email security and offers a valuable tool for combating the ongoing issue of email spam.

Keywords—Email spam detection, Machine learning, Natu- ral Language Processing (NLP), Naive Bayes classification, Text vectorization, Text classification.

I. INTRODUCTION

Email which stands for electronic-mail is a form digital communication in which people using electronic devices send and receive digital messages over the Internet.

Email was invented in the 1970s. Today it has become an integral part of modern communication and is used for various purposes, including personal communication, professional correspondence, marketing, and more. Development of email client software (like Outlook) and web browsers have made emails more popular than ever.[1]

Unsolicited and unwanted junk email, commonly known as spam, is sent out in large quantities to a broad and indiscriminate recipient list. The primary motive behind spam is usually commercial. It is often distributed on a massive scale through botnets, which are networks of compromised computers.

Spam emails pose potential risks. They may contain harmful links that can infect your computer with malware. It is crucial to refrain from clicking on any links in spam emails. Moreover, malicious spam messages often create a sense of urgency, prompting recipients to take immediate action.[2]

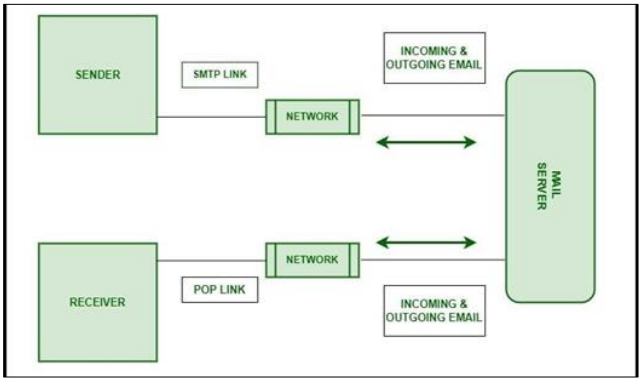

Fig. 1. Working of an email.

Some common types of Spam

- Commercial advertisements
- Antivirus warnings
- Email spoofing
- Sweepstakes winners
- Money scams
- Malware
- Phishing

NLP can be used to sense junk by analyzing text and metadata from emails and extracting features to distinguish spam from non-spam. This includes tokenizing the email text into words, sentences, or n-grams; removing stop words, punctuation, and other irrelevant tokens; normalizing the text by stemming, lemmatizing, or lowercasing; vectorizing the textwith bag-of-words, TF-IDF, or word embeddings; extracting sentiment, emotion, or tone from the text; identifying named entities, topics, or keywords from the text; parsing the text into syntactic or semantic structures; and generating new text or

summaries from the text. By capturing patterns, styles, and intentions of spam emails and comparing them to non-spam emails, NLP features can be very useful in spam detection.

This study employs the SMS Spam Collection dataset to distinguish between spam and non-spam emails. The dataset is multivariate and sourced from a single email account. It serves as the basis for the claim of diverse machinelearning techniques to identify and classify junk emails. The analysis is conducted using the WEKA tool, an open-source software developed in Java. WEKA offers a range of algorithms for data analysis and predictive modeling. Following the application of these algorithms, metrics such as precision, recall accuracy, score, and the Correctly Classified Instancesat ten-fold cross-validation are computed. The classifier with the highest accuracy and correctly classified instances is then determined.[3]

II. Literature Survey

The issue of detecting spam emails has garnered considerable attention from researchers, leading to the proposal of various approaches. This part reviews earlier effort on spam categorization using machine learning (ML) and deep learning approaches.

Srinivasan et al. [1] investigated the role of term inserting in deep learning for electronic mail junk detection. In termsof efficacy, the solution they suggested performed much betterthan typical email representation methods.

In another study, Soni [11] introduced a deep learning model called THEMIS, utilizing an enhanced Recurrent Convolutional Neural Network (RCNN) to identify phishing mails based on both email header and frame content at both characterand word levels. THEMIS achieved an accuracy of 99.84%, surpassing LSTM and CNN in the experimentation.

Hassanpur et al. [12] utilized the word2vec library to represent emails as vectors, thus avoiding the rule- based approach. These vector illustrations are fed into a neural network (NN) for knowledge.

The method has proven to be very effective, outperforming traditional machine learning algorithms with over 96% accuracy.

Egozi et al. [13] investigated the potential of Natural Language Processing (NLP) techniques in identifying phishingemails. By processing email samples, the researchers extracted various features such as word count, stop words, punctuation, and special features. These 26 features are applied to train a joint learning model based on support vector machine (SVM).

The model proved effective, successfully detecting over 80% of phishing mails and 95% of genuine emails.

Seth et al. [5] introduced an unique hybrid model that leverages Convolutional Neural Networks (CNN) to scrutinize both text and images in emails for spam detection. This innovative approach demonstrated an impressive accuracy rate of 98.87

Ezpeleta et al. [6] enhanced the precision of junk identification by incorporating a polarity score feature into Bayesian filtering classifiers. This feature, which mirrors the semantic content of emails, boosted the accuracy to an astounding 99.21%. Their work underscores the potential of email sentiment analysis in bolstering spam detection mechanisms.

J. Choi, et.al (2021) investigated an effective model for detecting spam [14]. Initially, the first Machine Learning (ML) filter was employed for analyzing all messages, allowing the authentic ones to pass, and assigning labels to the suspected ones. Afterward, an expert was introduced which aimed to analyze the flagged messages for improving the entire accuracy. The next procedure implemented a cost-based ML method for preventing the fatal error when a spam message was falsely recognized as an authentic one. In addition, the rapid spam trends were removed after deploying a module which facilitates in periodically updating the expert's analysis on the training dataset into the investigated model. An imbalanced dataset was executed to compute the investigated model in the experimentation. The results revealed that the investigated model yielded an accuracy of 92.8contrast to the existing methods. Moreover, the stability of this model was proved.

G. Andresini, et.al (2022) formulated a new method called EUPHORIA for classifying the reviews as spam and authentic [16]. The multi-view learning (MVL) was integrated with Deep Learning (DL) for attaining accuracy on the basisof huge information related to the content of reviews and the conduct of reviewers. Two real time lists created by Yelp.com – 'Hotel' and 'Restaurant' were employed for computing the formulated method in experiments. The findings confirmed that the formulated method performed more effectively with MVL while detecting review spam. Furthermore, the formulated method yielded an accuracy of 81.3dataset and 70.8% on the second one in contrast to the traditional methods.

Z. Zhang, et.al developed a new regularized ELM based algorithm known as Improved Incremental Fuzzy- kernel-regularized Extreme Learning Machine (I2FELM), to detect the spam in Twitter in a precise way [15]. The fuzzy weights were utilized for resolving an unbalanced data problem so that the accuracy was enhanced. Furthermore, the effectiveness of this approach was increased by employing Cholesky factorization, which did not need the square root or the composite kernel function. Thereafter, this algorithm focused on verifying the reasonable number of hidden nodes in an automatic way. The experimental outcomes exhibited the applicability of developed algorithm for effectively recognizing the balanced and unbalanced dataset. Additionally, based on few attributes, this algorithm was proved effective for detecting spam as compared to other methods.

Chhabra et al. [17] created Spam Sieving using Support Vector Machine over Enron Dataset by usingNonlinear SVM

classifier with varied kernel functions. Here, six datasets were evaluated, and the investigation of datasets with varying spam: ham ratios produced appropriate Recall and Precision Values.

TABLE I. LITERATURE SURVEY ON EMAIL SPAMDETECTION

Authors	Year	Methodology & Key Findings	Performance
Srinivasan et al. [10]	2020	Deep learning word embedding for email spam detection. Superior results ascompared to conventional procedures.	Not specified
Soni [11]	2018	THEMIS model with enhanced RCNN for phishing message identification. Accuracyof 99.84%, surpassing LSTM and CNN.	99.84%
Hassanpur et al. [12]	2018	Utilized word2vec for email vector representation. Input vectors into NN for learning. Accuracy exceeding 96%.	>96%
Egozi et al. [13]	2018	NLP techniques for phishing email identification. Extracted features for SVM- based ensemble learning. Over 80% of phishing mails and 95% of authentic mails were successfully recognized.	>80% (phishing), >95% (legitimate)
Seth et al. [5]	2017	Hybrid model using CNN for text and image analysis in emails. Accuracy rateof 98.87%.	98.87%
Ezpeleta et al. [6]	2016	Enhanced spam identification with a polarity score feature in Bayesian filteringclassifiers. Accuracy boosted to 99.21%.	99.21%
J. Choi, et.al [14]	2021	Model expert for analyzing flagged messages. Cost-based ML to prevent fatal errors. Rapid spam trend removal. Imbalanced dataset. Accuracy of 92.8%.	92.8%
G. Andresini, et.al [16]	2022	EUPHORIA method for classifying re- views as spam or authentic. MVL integrated with DL. Used Yelp.com datasets. Accuracy of 81.3% (initial) and 70.8% (second).	81.3% (initial), 70.8% (second)
Z. Zhang, et.al [15]	2020	I2FELM algorithm for spam detection in Twitter. Utilized fuzzy weights. Im- proved efficacy using Cholesky factorization. Automatic verification of hidden nodes. Applicable for balanced and unbalanced datasets.	Not specified
Chhabra et al. [17] [2010]	2010	Spam Filtering using SVM. Nonlinear SVM classifier with differentkernel functions on Enron Dataset. Considered 6 datasets. Satisfactory Recall and Precision Values.	Not specified

III. VISUALIZING OUR DATASET

Here we can see the distribution of the message length.

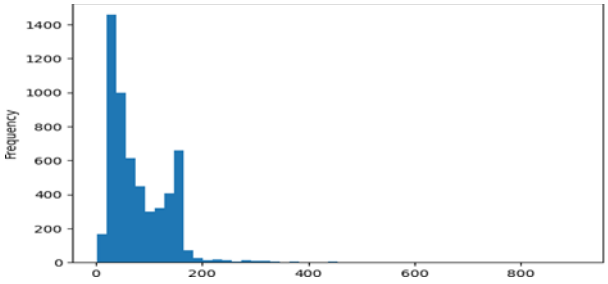

Fig. 2. Length of messages.

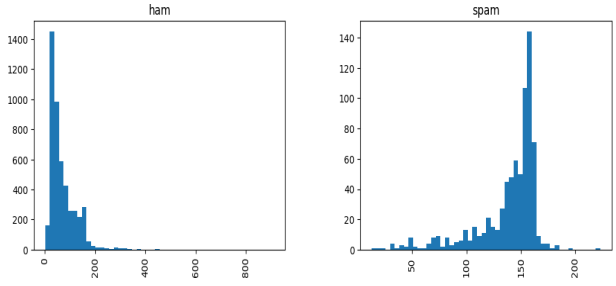

Fig. 3. Length ham and spam messages.

An important observation we found is that spam emails tendto contain more characters than real emails.

So, if you look here at the left where it's ham, we're looking at an average somewhere between zero and 200. It looks like it'scentered much closer to about 50 and if you look at spam text messages these are actually tend to be centered around 150.

So, as we've seen here just visually it looks like length is agood feature to distinguish a spam message versus a ham message.[5]

Normalization The messages in our dataset contain punctuations and some common words which are not useful in classifying messages as spam or ham. Firstly, we remove the punctuations from the messages because both spam and ham messages contain punctuation and therefore it cannot be used to distinguish whether a message is spam or ham(genuine).

```
nopunc = [char for char in mess if char
not in string.punctuation]
```

It is essential to eliminate stop words from the text messages. Stop words constitute a set of frequently used words in any language. For instance, in English, words like "the", "is," "and" are typical examples of stop words. The significance of removing stop words lies in the ability to concentrate on the more meaningful words in a given context.[2] In various applications, including Natural Language Processing (NLP) and text mining, the NLTK library's stop words package is commonly employed. This

tool aids in the removal of irrelevant words, enabling applications to prioritize and focus on the significant words in the text.

```
def text_process(mess):
    """
    Takes in a string of text,
    then performs the following:
    1. Remove all punctuation
    2. Remove all stopwords
    3. Returns a list of the cleaned text
    """
    # Check characters to see
    if they are in punctuation
    nopunc = [char for char in mess
        if char not in string.punctuation]

    # Join the characters again
    to form the string.
    nopunc = ''.join(nopunc)

    # Now just remove any stopwords
    return [word for word in
nopunc.split()
     if word.lower() not in
    stopwords.words('english')]
```

A. Vectorization

Presently, our objective is to convert the messages into vectors to ensure compatibility with SciKit Learn. This transformation will occur in three phases using bag-of-words model: Computing occurrence of every word in every message(known as term frequency). Allocate weights to the counts, diminishing the weight for tokens that appear frequently (termed as IDF). Adjust the vectors to have a magnitude of one, thereby disregarding the original text length.

The dimensionality of every vector will correspond to the count of different words in the collection of SMS.[3] Initially, we will utilize Count Vectorizer of SciKit Learn, amodel engineered to transform a set of documents into anarray depicting the number of tokens. In theory, this matrixcan be visualized as two-dimensional, where one dimension covers the entire vocabulary (one line for each word), and the other dimension pertains to the real documents, represented bycolumns, with each column denoting a unique text message.[6]

	Message 1	Message 2	...	Message 3
Word 1 Count	0	1	...	0
Word 2 Count	0	0	...	0
...	1	2	...	0
Word N Count	0	1	...	1

Fig. 4. Message vector.

Given the abundance of messages, it is anticipated that many words will exhibit zero counts for their presence in specific documents. Consequently, SciKit Learn is designed to yield a Sparse Matrix as output.

TF-IDF is a frequently employed weighting factor in the domains of information retrieval and text mining. This statis- tical metric is utilized to assess the relevance of a word in a document that is portion of a broader group or corpus. The importance of a word is directly related to its occurrence in the file, but this is counterbalance by the occurrence of the word in the entire set.[4]

One of the simplest ranking methods involves adding up the tied scores for each query term. The TF- weight typically consists of two components. The first component is the TF, computed by dividing the occurrence of a word in a file by the total word count in that document. The second component is theIDF, calculated by taking the log of the ratio of the total no. of documents in the set to the number of files containing the specific term.[7]

TF (Term Frequency): This measure assesses the frequency of a term's appearance in a file. Considering the variation in document lengths, a term may appear more often in lengthy documents as compared to short documents. Hence, to normalize, the term frequency is often divided by the length of document:

TF(x) = (Count of term x's appearances in a file)

/ (Total count of terms in the file).

IDF: This metric assesses the significance of a term. In the computation of TF, all terms are treated as equally significant. However, specific terms such as 'is', 'of', and 'that' might appear often however hold minimal significance.[9] To tackle this, our aim is to reduce the influence of common terns while amplifying the importance of infrequent ones, accomplished through the subsequent calculation:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Fig. 5. TF-IDF Weight Formula

Example: Suppose we have a document with the following text:

> "Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention."

Now, let's say we want to calculate the term frequency of the word "data" in this document. First, we need to count the total no. of occurrences of the word "data" occurs in the file. Here case, "data" appears twice. Next, we need to count

the total number of words in the file. In this case, there are 26 words.

B. Algorithm used

Naive Bayes Classifier

The Naive Bayes classifier serves as a probabilistic machine learning model designed for classification, predicting outcomesbased on the probability of an object. This classifier is particularly prevalent in text classification scenarios, dealing with high- dimensional training datasets. The fundamental principle underlying the classifier is rooted in the Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Fig. 6. Bayes Theorem.

Bayes' theorem allows us to calculate the likelihood of an incident A happening given that incident B has already occurred. The assumption of this model is the independence of the predictor variables or traits; This means that the existence of one property does not disturb the existence of the other.[8]

For instance, let's take a training dataset that includes weather data and a corresponding target variable 'Play' that signifies the probability of playing. The goal is to classify whetherplayers will participate in the activity based on the weather conditions.

Steps: Convert the dataset into a frequency table.

- Create a Likelihood table by determining probabilities.
- Utilize the Naive Bayesian equation to compute the posterior probability.
- Apply the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability becomes the predicted outcome.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Fig. 7. Working of Naive Bayes classifier.

Let's consider a simple example of weather prediction. Suppose wewant to predict whether it will be rainy given that it is a particular day of the week (say Monday). Here, 'Rainy' is the class label B and'Monday' is the feature A1. According to Bayes' theorem, the posterior probability P(Rainy|Monday) can be calculated as follows: P(Rainy|Monday)=P(MondayRainy)P(Rainy)/ P(Monday)

Here: P(Rainy|Monday) is the posterior probability we want to calculate. P(Monday|Rainy) is the likelihood, which is the probability of it being Monday given that it is Rainy. P(Rainy)is the prior possibility of it being Rainy. P(Monday) is the evidence, which is the total possibility of it being Monday. Remember, the Naive Bayes classifier makes a 'naive' assumption of independence among the features. This means that if we had more features (like humidity, wind speed, etc.), [18] we would assume that they all contribute independently to the probability of it being sunny, which simplifies the calculation of the likelihood term.

C. Performance of Algorithm Used

1) Classification Report

TABLE II. CLASSIFICATION RESULTS.

	precision	recall	f1-score	support
ham	1.00	0.96	0.98	1009
spam	0.72	1.00	0.83	106
accuracy			0.96	1115
macro avg	0.86	0.98	0.91	1115
weighted avg	0.97	0.97	0.97	1115

IV. CONCLUSION & FUTURE SCOPE

The expected outcome of this project is a highly accurate and adaptable email spam detection system that significantly reduces the prevalence of spam in users' inboxes, enhancing theoverall email experience while maintaining a low false-positiverate. Effective email spam detection is crucial for protecting user privacy, security, and productivity. A successful system willlead to a reduction in the time and resources wasted on managing spam and improve user confidence in email communication. In email security, future enhancement is crucial.

There is a wide possibility for future improvement because dynamic spam tactics constantly evolve, demanding an adaptable system. By balancing false positives and false negatives we can expect better results. Multilingual support is essential for a global user base. Real-time email detection is paramount fora seamless user experience. The landscape of email security improvement is driven by evolving spam tactics, precise false positive/negative management, multilingual support, and real- time detection for user-friendliness.

REFERENCES

[1] S. Srinivasan, V. Ravi, M. Alazab, S. Ketha, A.-Z. Ala'M, and S. K. Padannayil, "Spam emails detection based on distributed word embedding with deep learning" in Machine Intelligence and Big Data Analytics for Cyber- security Applications, Springer, 2021, pp. 161–189.

[2] C Valliyammai, B Guhan, D Manikandan, K Bhavani Venkata Karthik, K Saktheeswaran. "ReSD: Realtime Spam Detection for Social Media Textual Conversation", 2023 12th International Conference on Advanced Computing (ICoAC), 2023

[3] S. Jancy Sickory Daisy, A. Rijuvana Begum. "Email Spam Behavioral Sieving Technique using Hybrid Algorithm", 2023 7th International Conference on I- SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2023

[4] N. V. Smirnov, A. S. Trifonov. "Classification of Incoming Messages of the University Admission Campaign", 2023 International Russian Smart Industry Conference (SmartIndustryCon), 2023

[5] S. Seth and S. Biswas, "Multimodal spam classification using deep learning techniques," in 2017 13th Interna- tional Conference on SignalImage Technology Internet- Based Systems (SITIS), IEEE, 2017, pp. 346–349.

[6] E. Ezpeleta, U. Zurutuza, and J. M. G. Hidalgo, "Does sentiment analysis help in bayesian spam filtering?" In International Conference on Hybrid Artificial Intelligence Systems, Springer, 2016, pp. 79–90.

[7] A. Bibi, R. Latif, S. Khalid, W. Ahmed, R. A. Shabir, and T. Shahryar, "Spam mail scanning using machine learning algorithm.," JCP, vol. 15, no. 2, pp. 73–84, 2020.

[8] S. A. Saab, N. Mitri, and M. Awad, "Ham or spam? a comparative study for some content-based classification algorithms for email filtering," in MELECON 2014-2014 17th IEEE Mediterranean Electrotechnical Conference, IEEE, 2014, pp. 339–343.

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

[10] Choudhary, N., Jain, A. K. (2017). Towards Filtering of SMS Spam Messages Using Machine Learning Based Technique. In Advanced Informatics for Computing Re- search (pp. 18-30). Springer, Singapore.

[11] A. N. Soni, "Spam-e-mail-detection-using-advanced- deep-convolution-neuralnetwork-algorithms," JOURNAL FOR INNOVATIVE DEVELOPMENT IN PHARMA- CEUTICAL AND TECHNICAL SCIENCE, vol. 2, no. 5, pp. 74–80, 2019. R. Hassanpour, E. Dogdu, R. Choupani, O. Goker, and

[12] N. Nazli, "Phishing e-mail detection by using deep learning algorithms," in Proceedings of the ACMSE 2018 Conference, 2018, pp. 1–1.

[13] G. Egozi and R. Verma, "Phishing email detection using robust nlp techniques," in 2018 IEEE International Con- ference on Data Mining Workshops (ICDMW), IEEE, 2018, pp. 7–12.

[14] J. Choi and C. Jeon, "Cost-Based Heterogeneous Learn- ing Framework for Real-Time Spam Detection in Social Networks with Expert Decisions," in IEEE Access, vol. 9, pp. 103573- 103587, 2021

[15] Z. Zhang, R. Hou and J. Yang, "Detection of Social Network Spam Based on Improved Extreme Learning Machine," in IEEE Access, vol. 8, pp. 112003-112014, 2020

[16] G. Andresini, A. Iovine and A. Appice, "EUPHORIA: A neural multi-view approach to combine content and behavioral features in review spam detection", Journal of Computational Mathematics and Data Science, vol. 1, no. 4, pp. 371-378, 22 April 2022

[17] Chhabra, P., Wadhvani, R., & Shukla, S. (2010). Spam filtering using support vector machine.Special Issue IJCCT, 1(2), 3.

[18] Pramod Kumar Soni, Pallavi Jain, "Single Camera based Real Time Framework for Automated Fall Detection" in Second International Conference on Computer Science, Engineering and Applications (ICCSEA), 2022. FIGURES

[19] Fig 1 working of an email : GeeksforGeeks

[20] Fig 4 Message Vector https://www.turing.com/kb/guide-on- word-embeddings-in-nlp

[21] Fig 5 TF-IDF Weight Formula https://mungingdata.wordpress.com/2017/11/25/episode- 1-using-tf-idf-to-identify-the-signal-from-the-noise/

[22] Fig 6 bayes theorm https://towardsdatascience.com/what- is-bayes-rule-bb6598d8a2fd?gi=b1e02b694886

[23] Fig 7 working of na¨ıve bayes classifier - https://www.datacamp.com/tutorial/naive-bayes-scikit- learn