

Email Spam Detection - A Comparative Performance Analysis

Vaibhav Bhat (1107179) Anita Yadav (1117437) Sonal Yadav (1105500)

Abstract—In recent times, email management has become an important issue due to a plethora of spam, virus and promotional emails. These spam emails may cause harm to your computer while promotional and social emails are more prone to exploitation. To solve these issues providers have proposed many algorithms to classify our emails as spam and ham. The most commonly used algorithm is Naive Bayes which helps in classifying given emails as spam and ham. In this project we provide a comparison performance of all the available algorithm like KNN, Naive Bayes, Support Vector Machine for email classification along with a new approach of implementing neural networks for sentiment analysis to help us classify the emails. We have used Enron corpus dataset of spam and ham emails for classification along with preprocessing of data using NLTK is done for better results. The comparative analysis is governed basically on three main factors: Accuracy, Recall and Specificity.

I. INTRODUCTION

With an increasing popularity of communication using electronic mails, we end up with a huge number of emails in our inbox every day. Emails being an inexpensive and quick way of communication has led people to share a huge amount of information, be it personal, commercial or official data daily. Despite being an easy way of receiving information, email users end up missing important emails. This is due to not classifying emails properly, like spam, non-spam, promotions, social, etc. Moreover, a lot of websites ask for our email IDs before we can get-in and read the full articles, due to which we unknowingly sign-up for their newsletter updates leading to more unwanted emails. We usually end up receiving these spam emails either as a plain email or even with certain attachments. There are also cases where the email contains images with the banner/advertisement of the given spam company. Hence, classifying them not only according to the text in the email but also taking into consideration these images becomes pivotal. This gives a more accurate classification of whether the given email is a spam or not. There have been new advances in which spammers bypass the spam filters and prevent their mail or images from being detected as spam. This has led to create a pressure to develop more efficient algorithms to detect such spam emails. Though Naive bayes being one of the most common algorithms used for email classification, we propose various other algorithms too. In this project we propose a comparative analysis of various algorithms that can be used for classification of emails. The set of algorithms that we use for comparison are- KNN, Naive Bayes, SVM and RNN. Sentiment analysis are one of the most common features being used as an input to machine learning algorithms. Sentiment analysis using neural network is being applied to predict the mood of any content that has

been processed. This gives us a better view of what the content is trying to showcase. We have tried using a similar approach using RNN for classifying the email dataset as spam and ham.

II. LITERATURE REVIEW

The survey paper on Learning to classify discusses the process of classifying and organizing the emails through various email classification system. Other techniques are also present which attempts to predict the most probable action that would be taken on a particular email- delete, forward, auto-reply, etc. The different systems providing email classification treat the emails as a le of words and try to remove all the HTML tags if present in the message body. Some systems also consider the mail headers separately by creating separate feature set for header items. The process of converting words to their root form by removing all the suffixes, known as stemming, reduces the number of distinct words to be considered. The Porter Stemmer algorithm utilizes this stemming process and reduces the processing overhead to certain level. Many words in natural language occur with high frequency but have low information content, such as a, an, the, and most prepositions and conjunctions, and can be removed with the assumption that no serious loss of information will occur. Algorithms assign a numeric value for e.g. count of the number of times a word is repeated, to a word in the bag of words. A representative set of features is derived for classification from a set of messages where each email message is represented as a binary feature vector (using the value 0 to indicate the absence of a feature), or as a vector with frequency counts. After this feature extraction, the classifier is trained on sample data. There are different techniques utilized for classification, naive bayes used in several systems utilizes the word counts in the bags of words to calculate its probabilities. RIPPER, based on the IREP classifier, is another rule-based classifier that was initially used for email classification. TF-IDF (Term Frequency-Inverse Document Frequency) is another popular classification method developed for text classification. In Support Vector Machines (SVM) a binary feature vector is fed as an input the classifier, and the SVM develops a hyper-plane to separate the data points into two classes, where a class denotes spam or non-spam.

The paper on Spam Filtering with Naive Bayes Which Naive Bayes? [9] discusses various different versions of Naive Bayes, and compare them on various non-encoded data-sets, that contain ham messages of particular Enron users and fresh spam messages. An experimental procedure has been performed, that emulates the incremental training of personalized spam filters. Roc curves are plotted which are used to

compare the different versions of Naive Bayes with respect to true positives and true negatives. The different versions of Naive Bayes algorithms are - Multi-variate Bernoulli NB; Multinomial NB, TF attributes; Multinomial NB, Boolean attributes; Multi-variate Gauss NB and Flexible Bayes. The paper evaluates these different NB algorithms by considering a situation faced by a new user of a personalized learning-based spam filter, adopting an incremental retraining and evaluation procedure[9]. The ROC curves generated from the evaluation provided an interesting result - good performance of the two NB versions - FB and the Multinomial NB with Boolean attributes, - these versions collectively obtained the best results in the experiments conducted. Another observation to be considered is - its lower computational complexity at run time and its smoother trade-off between ham and spam recall, thus authors prefer the Multinomial NB with Boolean attributes over FB. The accuracy results were achieved with the largest attribute dataset (with 3000 attributes), however, the gain was not that significant when compared to smaller attribute sets, as they were cheaper in terms of computation [9].

In the paper on Text and image-based spam email classification [7], a brief introduction on how emails have become one of the most important way of communication and how classifying various emails in separate categories is difficult is discussed. As increase in email communication leads to a generation of loads of spam emails, the authors have proposed various algorithms to classify the emails as spam and non spam and comparison between the algorithms used in terms of precision is made. The most important part of their work is the use of preprocessed data, which helped improve the accuracy and gave better results compared to the old approaches like SVM. Techniques like black listing (creating a list of domains used by spammers) and white listing (list of trusted domains) were used, but which did not give accurate results as spammers started embedding images containing the spam text. The various algorithms like KNN, naive bayes and their own implementation of Reverse DBSCAN is used[7]. While implementing KNN choice of K was made on random trial and error method and proximity was calculated using the Euclidian distance, which gave them good accuracy with high complexity. While using Naive Bayes, combination of probabilities of all the important features was done, and closer the probability to 0, more likely the message was considered as non spam. Other than this the self implementation of DBSCAN, the algorithm would separate objects into cluster, by calculating the distances, our approach takes cluster of ham and spam already denied. So, when a mail comes its distance from each of this cluster is calculated[7]. Based on the results shown, the highest accuracy was received with preprocessed data in naive bayes followed by KNN and reverse DBSCAN. Based on the results and approach proposed, many algorithms gave good results along with some disadvantages of text filtering, as they were time consuming and the OCR used for image classification did not give perfect results as expected. Based on the papers reviewed, email classification being the most common mode of communication needs an improvement to detect spam emails which makes it easier for users to classify between important emails and spam ones. Different

approaches are proposed to accurately classify the emails, not only via text but also through images. Thus, to achieve the objective of increasing the accuracy of classifying spam emails, we are proposing to use the Multi variate Bernoulli NB technique along with focusing on implementing the future scope of detecting the viruses in the emails and also to classify promotional emails which makes life easier for the users.

In the paper on Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data [4] authors have evaluated machine learning models for sentiment analysis on text reviews created one language, on different other languages which do not have enough training data. Analysis of sentiments is an important part of understanding user behavior and perceptions about goods, locations or services. The search community has researched sentiment analysis for a long time, resulting in several sentiment-related tools, such as sentiment dictionaries, which can be used as features for machine learning models. These resources help increase sentiment analysis accuracies; however, they are highly language-dependent and require researchers to develop certain tools for the processing of each language. Authors have developed a portable model of sentiment analysis that uses no lexicons. The goal is to evaluate how well a generic model can be used to mine opinion in different languages where data is more limited than the language where the generic model is trained on[4]. To that end, authors built a training set that contains reviews from different domains in English (e.g., movie reviews, product reviews) and train a recurrent neural network (RNN) model to predict polarity of those reviews. Then focusing on a domain, by using the trained weights from the larger data and further training with data on a particular domain, a model is made which is specialised in that domain. To evaluate their usability of the sentiment analysis model, we test with non-English datasets. First the test set is translated into English and a pre-trained model is used to score polarity in the translated text. In this way, the proposed approach eliminates the need to train language dependent models, use of sentiment lexicons and word embeddings for each language [4]. The contributions of the research paper are:

- 1) A flexible solution that reuses a template trained in one language in other languages using machine translation
- 2) An RNN-based approach to remove the collection of features and resource criteria for sentiment analysis
- 3) A technique that performs multi-lingual sentiment analysis baselines statistically significantly when data is small.

III. METHODOLOGY AND JUSTIFICATION

Email logs have been considered as a useful resource for research in fields like link analysis, social network analysis and textual analysis and spam detection. However, lack of real-life benchmark data has been an obstacle for studying the problem and evaluating the results. The Enron email dataset is a collection of email messages from the Enron corporation. This dataset proves to be a perfect test bed for testing the effectiveness for spam detection as it is very similar to the kind of data collected for fraud detection.

A. Email Dataset Used

The Enron email dataset is a collection of email messages from the Enron corporation.[12] This dataset proves to be a perfect test bed for testing the effectiveness for spam detection as it is very similar to the kind of data collected for fraud detection. There a total of 33716 email messages belonging to 158 users. The emails are classified into group of 16545 emails as HAM and 17171 SPAM emails. The dataset is further divided into subsets where each folder consists of spam and ham emails. Each message present in the folders contains the senders and the receiver email address, date and time, subject, body, text and some other email specific technical details.

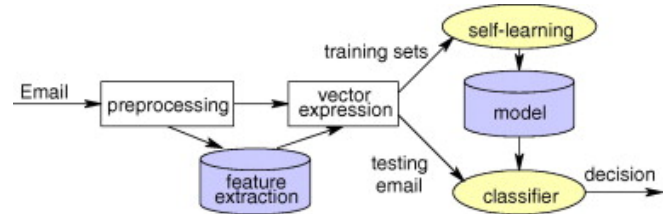
B. Pre-Processing

For efficient processing of our algorithms, we pre process the raw data obtained from the Enron email dataset. After complete analysis of the dataset, we discovered that some emails have junk data, some are blank, and some are duplicated. On further observation we discovered that some emails consist of threads, which are sequence of messages on a topic, and consists of certain repetitive words, which we plan upon using as a one of the features in our feature vector. To achieve the objective of classifying emails the dataset is divided into train 70% of our data and test the remaining 30%. For setting up the feature vector we plan to make a dictionary and select common features like, the most commonly used words, some specific email ids and large use of special characters in the emails. Based on all these common features, we will make a feature vector which will be used for implementing the algorithms proposed. Some steps involved in preprocessing are:

- Python NLTK (Natural Language Toolkit) is used to perform pre-processing.
- Preliminary preprocessing:
 - Removing footer, duplicate/blank email
 - Retaining the subject as it is used for detecting spam content
- Tokenize - We create "tokens" from each word in the email by removing punctuation.
- Remove meaningless words - Stop-words do not provide meaningful information to the classifier, and they increase dimensionality of feature matrix.
- Stem - Similar words are converted to its stem in order to form a better feature matrix. This allows words with similar meanings to be treated the same. Each stem is placed into our "bag of words", which is just a list of every stem used in the dataset.
- Create feature matrix - After creating the "bag of words" from all of the stems, we create a feature matrix. The feature matrix is created such that j column contains all the words in ith mail.

Various efficient spam detection algorithms are available having different levels of accuracy and performance. However, with frequent changes in spam mails patterns it becomes difficult for these techniques to detect spam mails. Thus, there is a need to keep finding enhanced and efficient algorithms to classify the emails. In the project we will implement and

evaluate the performance of 4 Machine Learning algorithms- Naive Bayes, Long short-term memory (LSTM) Neural Network, Support Sector Machine (SVM) and k-Nearest Neighbors (kNN) - commonly used for classification.



Following sub-sections discuss about the implemented algorithms and the last sub-section talks about the performance evaluation metric used to compare the different techniques.

C. Naive Bayes

Naive Bayes algorithm is a popular text classification technique used in various applications. It relies on Bayes Rule and classifies each object by looking at all of its features individually. This means, the classifier assumes that the possibility of occurrence of a feature in a class is completely unrelated to the presence of any other feature. They use the bag-of-words feature model to classify the text to identify the classes the text belongs to. Every word will have a probability of occurring in a spam or a non-spam email. Naive Bayes algorithm can be used to compute the probability that an email with a set of words in it belongs to which category. The posterior probability of the object is calculated for each feature and then these probabilities are multiplied together to get a final probability. This probability is calculated for the other class as well. With these probabilities of different classes, it can be predicted that the object belongs to the class with the highest probability. In our problem, the object is an email and the features are the unique words in those emails. Thus, a posterior probability is calculated for each unique word in the email. The classifier must be trained first to find the words which occur the most in a spam mail. For instance, Earn\$ frequently appears in the spam mails and will rarely be found in other emails. The filter will be trained to identify these kinds of words to have a very high spam probability and a very low spam probability for words seen only in legitimate emails (like names of family members and friends). Once the filter is trained, it is tested against the dataset by finding the word probabilities and using them to compute the probability that an email with a set of words in it, is a spam or not. Each key word in an email helps in determining the probability of it being a spam. This contribution is known as the posterior probability and is computed via Bayes' theorem. Thus, the email's spam probability is calculated by considering all the words in the email, and if the total exceeds a certain threshold, the classifier will mark the email as a spam.

With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by a multinomial where $p(i)$ is the probability that event I occurs (or K such multinomials in the multiclass case). A feature vector $x = (x_1, \dots, x_n)$ is then a histogram, with x_i counting the number of time event I was observed

in a particular instance. This is the event model typically used for document classification, with events representing the occurrence of a word in a single document (see bag of words assumption). The likelihood of observing a histogram \mathbf{x} is given by:

$$p(\mathbf{x} | C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p_{ki}^{x_i}$$

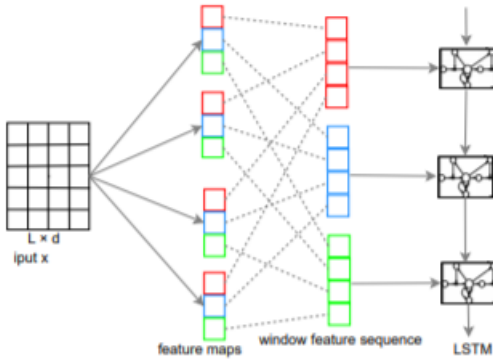
The multinomial naive Bayes classifier becomes a linear classifier when expressed in log-space

$$\begin{aligned} \log p(C_k | \mathbf{x}) &\propto \log \left(p(C_k) \prod_{i=1}^n p_{ki}^{x_i} \right) \\ &= \log p(C_k) + \sum_{i=1}^n x_i \cdot \log p_{ki} \\ &= b + \mathbf{w}_k^\top \mathbf{x} \end{aligned}$$

where $b = \log p(C_k)$ and $w_{ki} = \log p_{ki}$.

D. Long short-term memory (LSTM)

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) used in the field of deep learning. RNN is an accurate neural network in theory, however, it has difficulty in remembering long term memories. Basically, when the distance between the current layer and the previous layer increases, the memory becomes weak. In these cases, the gradients either become too small or too large (the vanishing/exploding problem). To overcome this issue, we use LSTM. LSTM network is able to sustain the memory for longer periods of time as compared to the RNNs.

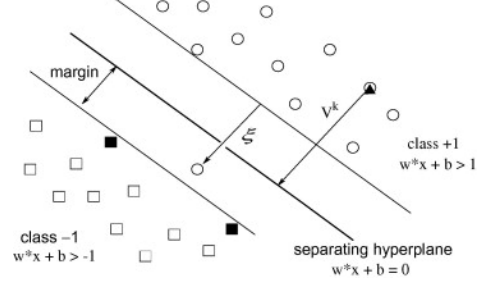


The libraries for spam detection using LSTM includes Keras and sklearn. In order to classify the emails as spam or ham, the basic idea is to perform text analysis on the given data using neural networks. We then load the data we get from the Enron Corpus (the database which we have used for the project). This data contains the emails with proper labels as spam or not spam. This is imported as the training data which is then used as test data as well using cross validation. We would then apply the Embedded layer over input data. This step is done before we add the LSTM layer into the Keras sequential model. Finally, we will train the model and test the

accuracy, precision, recall and the F1-score which would be used to compare with other algorithms.

E. Support Sector Machine (SVM)

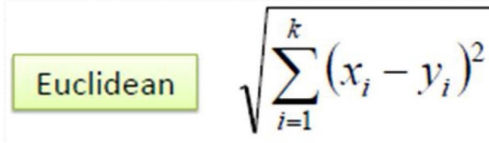
A support vector machine (or SVM) is a machine learning algorithm that analyzes data for classification and regression analysis. SVM is a supervised learning method that looks at data and classifies it into one of two categories using a hyper plan. The aim is to search out a hyper plan, which may increase the margin between the two classes. Our email spam detection problem is a binary classification problem where the emails are considered as the data and the output is either spam (or 1) and not spam (or 0). Hence, we can use an SVM classifier for our email spam detection.



For implementing SVM classifier for spam email detection, we first have to extract the useful features from the whole data after the preprocessing step. The result of this feature extraction is a feature vector which is used as an input. Here, in SVM, these are called as weighted features. We then convert these weighted features into a specific format so that we are able to use the SVM classifier function directly on the features. This conversion to a specific format is called feature representation. We represent these weighted features as Vector Space Model (VSM). VSM represents emails as vectors and all features are binary, i.e. $x=1$ if the feature is available in the email or 0 otherwise. This would comprise our data for implementing SVM. We will use cross-validation to divide the data into training and test sets for calculating the accuracy, precision, recall and F1-score which will be used for comparing them with the results of other algorithms.

F. k-Nearest Neighbors (kNN)

K nearest neighbour (kNN) is a very simple yet very powerful classification technique in machine learning if used at right place. In pattern recognition, kNN algorithm can be used for classification and regression. However, we see the most widespread application of kNN for classification purpose. In the training phase, only the feature vectors and class labels are stored. Now when we get an unlabelled data, the algorithm basically finds the k-nearest-neighbours based on the feature vectors stored beforehand. The distance metric used to find the k-nearest neighbours is dependant on a lot of variables, however using the basic trial and error or cross validation is considered to be the best way. Some of the well-known distance metrics include Euclidean distance, Minkowski distance and Manhattan distance.



The diagram shows a yellow box labeled "Euclidean" next to the mathematical formula for Euclidean distance: $\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$.

The value of k is also decided by the trial and error method, however certain guidelines can be followed in order to find the best k for the given dataset. After finding these k -nearest neighbours, the unlabelled data is labelled according to the majority of these nearest neighbours that were found. The problem to detect whether the given email is spam or ham is a classification problem. As a result, we can apply the powerful kNN algorithm to solve our problem. However, in kNN, it doesn't make sense to pass the data as a string/text as finding distances in text would be meaningless. As a result, we have to preprocess the data to get certain features (the feature vector extracted) which can be used to perform kNN on the given Enron Corpus dataset. The sklearn library is used to implement the kNN algorithm in python. The feature vector along with the respective labels is passed as the training data. The labels are obtained from the Enron Corpus dataset which classifies the data as spam or not spam. The Euclidean distance will be used as the distance metric to find the nearest neighbours. Furthermore, using cross validation, we find the accuracy, precision, recall and F1-score to compare the algorithm with the other mentioned algorithms.

G. Evaluation Metric

The main goal of our classification is to make an evaluation matrix which clearly evaluates the performance of the various models that we intend to implement. For the comparison we plan to use the 3 pillars of classification which are Accuracy: which defines how comfortable the model is with detecting the positive and negative class; Precision: which tells about the success probability of making correct positive class classification; Recall: which defines how sensitive the model is towards identifying the positive class. In addition to the 3 factors we will also look upon the F1 score of each model which defines the weighted average of precision and recall, considering both false positive and false negative. Our investigation and research on the papers that implemented email classification resulted in showing average performance by models implementing KNN and SVM, with some constraints and high performance displayed by Naive Bayes and Recurrent Neural Networks. We also intend on achieving similar results and precision close to the ones achieved in the researches. We measure various parameters for comparative analysis of the different algorithms:

- **Specificity:** It measures the proportion of actual positives that are correctly identified as such.
 - $\text{Specificity} = [\text{TN}] / [\text{TN} + \text{FP}]$
- **Accuracy:** Accuracy shows us how comfortable the model is with detecting the positive and negative class. Its computed by the sum of True Positives and True Negatives divided by the total population.
 - $\text{Accuracy} = [\text{TP} + \text{TN}] / \text{Total Mail}$

- **Recall:** It depicts how sensitive the model is towards recognizing the positive class. Its calculated as number of True Positives divided by the sum of True Positives and False Negatives.

- $\text{Recall} = \text{TP} / [\text{TP} + \text{FN}]$

Terms Used:

- True Positive(TP): Spam files correctly classified
- True Negative(TN): Ham files correctly classified
- False Positive(FP): Ham incorrectly classified as spam
- False Negative(FN): Spam incorrectly classified as ham

IV. RESULTS

We performed 5-fold cross validation to measure the performance of the four different algorithms and compare them to learn which of them is better suited for email spam/non-spam classification. We achieved our best accuracy from SVM algorithm that is close to 97%.

As shown in tables below we achieved similar results in SVM and Naive Bayes by performing 5-cross validation and executing without n-fold cross validation by splitting the data in train and test. SVM accuracy without n-fold cross validation gave us an accuracy of around 97% which remains same when we performed 5-fold cross validation on data 97%. Similar is the case with Naive Bayes algorithm which gave accuracy around 96% in both cases with and without n-fold cross validation. However, we saw major difference in accuracy of kNN when executed without 5-fold cross validation. It increased drastically from 68% to 83% without cross validation. However, RNN saw a significant fall in its accuracy without cross validation, it went down to around 30% as compared to accuracy with cross validation, i.e. 61%.

5-fold cross validation results

Algorithms	Accuracy
kNN	0.68
SVM	0.97
Naive Bayes	0.96
RNN	0.61

The other parameters calculated for comparative analysis of the four different algorithms were Specificity and Recall. Specificity shows the ratio of correctly predicted ham emails out of the total ham emails whereas Recall shows the ratio of the correctly predicted spam emails out of the total spam emails. As shown in the second table- the specificity of kNN and SVM was same and slightly better than that of Naive Bayes. Whereas, for recall SVM and Naive Bayes performed somewhat better than kNN, with SVM achieving the best value in this category as well.

Train-Test Data split results

Algorithms	Accuracy	Specificity	Recall
kNN	0.83	0.98	0.91
SVM	0.97	0.98	0.94
Naive Bayes	0.96	0.97	0.93
RNN	0.30	-	-

V. CONCLUSION

In this project, we applied four different algorithms to detect spam emails one of which was one of the novel method for email spam detection. We tried to classify the emails in the enron corpus dataset as spam or ham emails. Although we expected that our novel approach using recurrent neural networks would provide better results as compared to the usual methods, it certainly does have some shortcomings which could not be solved to obtain a better accuracy. Overall, by analysing the enron corpus dataset for email spam detection it is clear that the traditional classification algorithms like SVM or Naive Bayes provide way better classification over recurrent neural networks.

VI. FUTURE WORK

It is easier for a human being to identify whether an email is spam or not just by looking at it. Hence, we do expect that neural networks should provide a better classification surpassing the traditional classification methods. In future, Artificial Neural Network (ANN) can be used for better classification with the same idea used in a chatbots for text classification. It might also be possible that ANN could solve certain mentioned issues in recurrent neural networks for better classification.

VII. ACKNOWLEDGEMENT

- Sonal Yadav - kNN, Naive Bayes, Data Preprocessing
- Vaibhav Bhat - RNN, kNN
- Anita Yadav - SVM, Data Preprocessing, RNN

REFERENCES

- [1] A. A. Alurkar, S. B. Ranade, S. V. Joshi, S. S. Ranade, P. A. Sonewar, P. N. Mahalle, and A. V. Deshpande, "A proposed data science approach for email spam classification using machine learning techniques," *Joint 13th CTTE and 10th CMI Conference on Internet of Things - Business Models, Users, and Networks*, vol. 2018-January, pp. 1–5, 2017.
- [2] I. Androutsopoulos, J. Koutsias, K. V. Chandrinou, and C. D. Spyropoulos, "Experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages," *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, pp. 160–167, 2000.
- [3] S. Bin Abd Razak and A. F. Bin Mohamad, "Identification of spam email based on information from email header," *International Conference on Intelligent Systems Design and Applications, ISDA*, pp. 347–353, 2014.
- [4] E. F. Can, A. Ezen-Can, and F. Can, "Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data," 2018. [Online]. Available: <http://arxiv.org/abs/1806.04511>
- [5] L. Firt, C. Lemnaru, and R. Potolea, "Spam detection filter using KNN algorithm and resampling," *Proceedings - 2010 IEEE 6th International Conference on Intelligent Computer Communication and Processing, ICCP10*, pp. 27–33, 2010.
- [6] V. Goel, A. K. Gupta, and N. Kumar, "Sentiment analysis of multilingual twitter data using natural language processing," *Proceedings - 2018 8th International Conference on Communication Systems and Network Technologies, CSNT 2018*, pp. 208–212, 2018.
- [7] A. Harisinghaney, A. Dixit, S. Gupta, and A. Arora, "Text and image based spam email classification using KNN, Naive Bayes and Reverse DBSCAN algorithm," *ICROIT 2014 - Proceedings of the 2014 International Conference on Reliability, Optimization and Information Technology*, pp. 153–155, 2014.
- [8] J. Kaur and B. K. Sidhu, "Sentiment Analysis Based on Deep Learning Approaches," *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICCC 2018*, no. Icccs, pp. 1496–1500, 2019.
- [9] V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with Naive Bayes - Which Naive Bayes?" *3rd Conference on Email and Anti-Spam - Proceedings, CEAS 2006*, 2006.
- [10] N. C. Poornima, "An implementation framework for real-time spam detection in Twitter," vol. 5, no. 2, pp. 1752–1755, 2019.
- [11] S. B. Rathod and T. M. Pattewar, "Content based spam detection in email using Bayesian classifier," *2015 International Conference on Communication and Signal Processing, ICCSP 2015*, pp. 1257–1261, 2015.
- [12] J. Shetty and J. Adibi, "The Enron Email Dataset Database Schema and Brief Statistical Report 1 University of Southern California USC Information Sciences Institute Figure 1 : Enron Database Schema Description of the tables in the MySQL database :," p. 7.
- [13] D. Tran, W. Ma, D. Sharma, and T. Nguyen, "Possibility theory-based approach to spam email detection," *Proceedings - 2007 IEEE International Conference on Granular Computing, GrC 2007*, no. 1, pp. 571–575, 2007.
- [14] C. H. Wu, "Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks," *Expert Systems with Applications*, vol. 36, no. 3 PART 1, pp. 4321–4330, 2009. [Online]. Available: <http://dx.doi.org/10.1016/j.eswa.2008.03.002>
- [15] B. Yu and Z. ben Xu, "A comparative study for content-based dynamic spam classification using four machine learning algorithms," *Knowledge-Based Systems*, vol. 21, no. 4, pp. 355–362, 2008.
- [16] B. Yu and D. hua Zhu, "Combining neural networks and semantic feature space for email classification," *Knowledge-Based Systems*, vol. 22, no. 5, pp. 376–381, 2009. [Online]. Available: <http://dx.doi.org/10.1016/j.knosys.2009.02.009>