International Journal of Advance Research in Science and Engineering
Volume No. 13, Issue No. 04, April 2024
www.ijarse.com
IJARSE
ISSN 2319 - 8354

Inbox IQ: An Automated Email Reply System

Shweta Kambare[1], Aabha Rudrabhate[2], Srinath Divate[3], Yatharth Thakare[4],

Anurag Tekam[5], Tapasvi Taktode[6]

Department of Artificial Intelligence & Data Science, Vishwakarma Institute Of Technology Pune, India

shweta.kambare@vit.edu, aabha.rudrabhate21@vit.edu, v.srinath211@vit.edu

yatharth.thakare211@vit.edu, anurag.tekam21@vit.edu, tapasvi.taktode21@vit.edu

Abstract— the ever-growing tide of emails threatens to drown us all in information overload. This paper proposes a novel machine learning system to conquer this challenge, empowering individuals and organizations to reclaim control of their inboxes. The proposed system tackles the email deluge through three key innovations: smart prioritization utilizing TF-IDF and Doc2Vec to identify crucial messages, effortless autoreplies generated by both keyword matching and advanced sequence-to-sequence models, and seamless Gmail integration through an API extension for minimal disruption. This intelligent solution, demonstrably outperforming existing tools, paves the way for a future where email management is efficient, effortless, and empowering.

Keywords— Doc2Vec, Naive Bayes, Priority Tokens for automated emails ,Tf-Idf.

I. INTRODUCTION

The digital age has fostered unprecedented connectivity, with email emerging as a cornerstone of personal and professional communication. However, this convenience comes at the cost of information overload. The average office worker receives and sends over 120 emails daily, translating to significant time spent reading, organizing, and responding. Imagine sifting through an ever-growing pile of paper letters, manually prioritizing each one and crafting personalized responses - that's the reality for many email users today. These basic functionalities fail to adapt to the complexities of modern communication, leaving users drowning in a sea of information. An estimated 13 hours per week are consumed by email management, hindering productivity and impacting work-life balance. Beyond individual burden, email overload poses challenges for organizations. Inefficient email management translates to lost revenue, decreased employee engagement, and potential security risks. Finding relevant information amidst a deluge of emails becomes arduous, while manually prioritizing and crafting responses leads to inconsistencies and delays. Traditional email clients offer limited solutions, often relying on rudimentary sorting and filtering mechanisms that fail to adapt to the complexities of modern communication.

Therefore, developing intelligent and efficient email management systems becomes crucial. Machine learning presents a promising avenue for addressing this challenge. This paper introduces a Smart Email Prioritization, and Automatic Reply Generation System that leverages machine learning techniques to transform email management.

The proposed system tackles the problem from three key angles:

Prioritization: Utilizing TF-IDF scoring on email tokens, the proposed system goes beyond simple keyword matching to identify crucial information and assign priority based on its relative importance within the broader email landscape. This nuanced approach ensures that critical messages receive timely attention, boosting productivity and streamlining workflows.

Automatic Reply Generation: The proposed system explores two avenues: keyword matching for highly structured responses and sequence-to-sequence models for generating more natural and context-aware replies. This allows us to address a wider range of email types and provide responses that are both relevant and tailored to the specific content and tone of the communication.

Seamless Integration: Recognizing the importance of user experience, the proposed is designed as an API extension for Gmail. This ensures seamless integration with existing workflows, minimizing disruption and enabling easy adoption by individuals and organizations alike.

This paper delves deeper into the technical aspects of our system, showcasing its architecture, methodology, and evaluation approaches. Compelling evidence is also presented of its effectiveness compared to existing solutions, demonstrating how machine learning can empower individuals and organizations to navigate the ever-growing sea of email communication with greater efficiency and ease.

II. LITERATURE REVIEW

[1] This work presents an improvement to handle email handling problems by integrating efficient information collection for email categorization and producing pertinent dictionaries when emails change in kind and volume. The research's open premise is that text messaging serves as the fundamental idea behind fan emails. It has been noted that the Automated Email Reply algorithm performs better when NLP techniques are applied, improving its capacity to classify and produce email responses using probabilistic methodologies with the fewest possible errors. An enhanced algorithm employs machine learning approaches to automate its functions and help email users who struggle with managing a large volume of different types of emails.

Secondly,[2] Eric Horvitz, Andy Jacobs, David Hovel used machine learning techniques to assign email a low or high priority. The approach developed ROC curves for the SVM model while working on it. Prototypes of the PRIORITIES systems, which use Microsoft Outlook for email and scheduling, were showcased.

In Personalized Email Recommender System Based on User Actions [3] the approach suggests utilizing statistical techniques and user behaviors to create an email recommender system. Authors approach the issue as a classification with multiple classes where each class represents a suggested action from the user to an email, as opposed to a two-class categorization with Spam and Ham. The most often performed actions are read, delete, and reply. To test the framework, an experiment is carried out wherein a Naïve Bayesian classifier with varying thresholds is used to assess the relationship between the performance and the quantity of features. An encouraging outcome with good forecast accuracy is shown by the experiment.

[4] It is suggested to employ an automated process that builds each user's personal profile by automatically extracting the features of each communication they receive and allocating a priority for each one. By keeping track of the messages; the person sends and receives, it creates the profile. Three elements make up the user's personal profile: the subjects retrieved from the message's body, the sender and recipient(s) identified in the email header. The suggested approach parses through the message body and extracts information about the message's urgency and type. Using the weights of each characteristic as established by multiple-regression analysis, the priority of a message is computed as the weighted sum of those characteristics. An experiment shows how the suggested approach might be used in real-world scenarios, such filtering or ranking the numerous messages that are received.
[5] Authors concentrate on three areas: 1) They look at using regression and classification for expressing the ordinal relations amongst the priority levels. 2) To identify user groups and acquire rich information that depict social roles from the perspective of a specific user, they examine personal social networks. 3) They also created a semi-supervised (transudative) system for learning that uses user nodes and messages in a personal email network to spread priority labels from training scenarios to test cases. By combining these techniques, they provide an improved vector representation and a better modelling priority for every new email message.
In the research, Smart Reply: Automated Response Suggestion for Email [6]; researchers suggest and study a new end-to-end approach called Smart Reply that generates concise email responses automatically. With a single tap on a mobile device, it provides semantically varied ideas that may be utilized as whole email responses. Currently, 10% of all mobile responses are assisted by the technology, which is used in Gmail's Inbox. It is intended to process hundreds of millions of messages every day at a very high throughput. The system makes use of cutting-edge, extensive deep learning.
The study “Customized Automated Email Response Bot Using Machine Learning and Robotic Process Automation (2019)” [7] proposes a system for producing intelligent responses to specific email classifications. The goal of the proposed system is to create an email management framework that can automatically identify and respond to certain categories of emails. Support Vector Machine Algorithm is used by the system to classify emails according to their content. An important factor in intelligent email response is the quality of the content analysis, since sending the wrong kind of email to a client can result in unanticipated consequences. Therefore, it is crucial to evaluate the system's viability on predefined email classes before carefully training the intended system and subsequently expanding the number of classes and appropriate templates.
[8] In this work, authors create a prototype for an intelligent email client that assists in email replying by offering a list of responses extracted from previously answered emails. The degree of similarity among the proposed responses determines their ranking. They have assessed real-world email data to determine that reuse is feasible, albeit with cautious retrieval techniques. In order to address the problem by utilizing previously composed responses from the past replies saved in the case base, they construct and analyze a case-based reasoning strategy. Applying text processing and semantic analysis approaches, they develop a retrieval system that detects similar situations beyond the matching. They use and cross-evaluate text analysis techniques, including synonym expansion and lexical analysis, to optimize retrieval. The results of their evaluation suggest that synonym expansion may increase the likelihood of obtaining a more relevant match, even at lower rankings. They assess their prototype by looking at the index size, processing time elapsed, and quality of retrieval results. In the study,

a unique Smart Email Client prototype built on the Case-Based Reasoning (CBR) framework for problem-solving was described. The approach developed a retrieval system that uses text processing along with semantic analysis approaches to locate related situations beyond exact matching. Depending on the entering query, the algorithm cleverly retrieves related previous queries and their corresponding responses. Researchers conducted a thorough evaluation of their prototype using three main criteria: the index size, processing time, and retrieval result quality. The dataset consisted of the mailbox data of two Enron employees: Germany-c and Farmer-d. It demonstrated that the outcomes obtained across all three criteria are encouraging.

[9] This research uses machine learning approaches to identify spam. The paper describes algorithms that use machine learning and applies them to various data sets. The optimal algorithm is chosen for email spam detection based on its best precision and accuracy.

[10] As part of the research, Shreyasi Sinha, Isha Ghosh, and Suresh Chandra Satapathy have created a BP as well as a BP+M framework to classify spam and determine classification accuracy. After comparing the two models, researchers are able to say that the BP+M model uses fewer epochs to get results that are either equivalent or better than the BP model. While backpropagation (BP), backpropagation with momentum (BP+M), two of the most popular and extensively studied classical and state-of-the-art learning algorithms, are known to frequently become stuck in local optima.

A. Pre-Processing

The conversion of this text data to an attributes list and into a CSV file posed the biggest hurdle. As this dataset was huge, we had to select specific users to create a compact dataset which would be suitable for computation and easy analysis. We created data frames from the csv generated and wrote multiple queries to pick top user interactions with considerable data. The results reveal Badeer as a suitable employee whose emails could be considered. Figure [1] illustrates the employees with the most interaction with our selected employee. All the .mbox files for the selected employee were traversed one by one and each of these files were scraped for data.

III. DATASET

Creating a model that can automatically generate responses based on previous messages requires feeding it with emails for training. The Enron Dataset is utilized as the raw dataset for building the model. In real-time, users can download their .mbox file to train the model, enabling it to generate responses specifically from the user's emails. Enron Corporation, an American energy and services company, released all its mail conversations after shutting down. The dataset consists of emails exchanged between employees and support requests from the company's customers. This dataset is a database of over 600,000 emails generated by 158 employees of the Enron Corporation Given the potential large number of emails during the model training process, parsing the emails can take a considerable amount of time. To mitigate this, the process of parsing the email archive and feeding it into the model is introduced during the initial model creation. Subsequently,

When new emails arrive, they will be added to the corpus after being compared.

IV. METHODOLOGY

Implementing the given system can be divides in below major steps:

1. Pre-Processing.
2. Response system implementation.
3. Priority Token Generator and Email ranking.
4. Mail response automation.

Fig. 1. Rank of user interaction with Badeer.

Emails needed to be sorted through and extracted from data. In order to detect the expression throughout all of the emails, we had to identify a general pattern. It was necessary to combine several different date formats into one. The dates were formatted as Sun., Dec. 30, 2001, 10:19:42 -0800 (PST),

Nov. 27, 2001, and 27 Nov. 2001. Every one of these several date formats was combined into a single datetime format. We also had to disregard the thread information in order to extract the features. Additionally, punctuation and stop word were eliminated.

The response generation algorithm required an annotated email feature list. Delete, reply, and thread are the tags included in the main recommendation. Delete indicates that the communication is either no longer pertinent or that some elements are outside the user's purview. Reply indicates an action item in the message must be completed by the user. Lastly, a thread email is one that is sent to the user's inbox without requiring any special action from them. The cleaned data had these features, To, From, Subject, Content, Date, and filename. All the mails in the dataset were annotated. To improve the information in the dataset three key augmentation techniques—Lemmatization, Stemming, and POS Tagging— were employed. The technique of stemming involves deleting the affixes from a given word to determine the base word. Eaten, eating, eats, for instance, has the stem eat. To obtain the word stems, we utilized Porter Stemmer from nltk.stemmer. Lemmatization is the process of identifying a term's base word. For example, ‘better’ has ‘good’ as the lemma. To obtain the lemma for each word, we utilized WordNet Lemmatizer from nltk.lemmatize. Lastly, context-relevant bits of speech are a crucial piece of information that will improve the model's accuracy. Therefore, we added POS tags to each word and concatenated them with

a delimiter ('/'). NLTK.POSTAG to obtain each word's POS. A clean CSV file was obtained once this preprocessing for Badeer, our target employee, was finished. The response generation system can be further implemented with the help of this file.

B. Response system implementation

Two models have been implemented to determine the type of response: the Doc2Vec sentence embedding model and the Term Frequency-Inverse document frequency (Tf-Idf) model. Term frequency-inverse document frequency is a numerical statistic intended to reflect how important a word is to a document in a collection or corpus. Tf-Idf generates a count vector matrix by taking into account the attributes From, To, Email date, Subject, and email content. The retrieved features were sent into the Count Vectorizer, sklearn feature extraction tool, after being concatenated. Next, the sklearn feature extraction package's Tf-Idf transformer was used to convert the count matrix to TF-IDF. The corpus was given the highest weights because it had more words than the other features. The accuracy and correctness of several classification models, including K Nearest Neighbor, Random Forest, Decision Trees, and Naive Bayes, were evaluated. The smallest sample size prevented Naive Bayes from classifying the delete responses, while having the best accuracy. Consequently, the requirement for more robust weighted sentence embedding model Doc2Vec.

Doc2Vec or Paragraph Vector allows documents to be represented as vectors. Each document in the high-dimensional space is mapped to a fixed-length vector. Similar texts are mapped to adjacent spots in the vector space thanks to the way the vectors are learned. This allows us to carry out operations like document categorization, grouping, and similarity analysis by comparing documents according to their vector representation. The individual tagged documents for the fields From, To, Email date, Subject, and Content are generated. Each feature had its own vocabulary built for it, which was then utilized to build the vectors for the feature model that was specified. Since SVM (Support Vector Machine) works well with vectors, it was employed with radial kernel bias. The regularization factor was adjusted for hyperparameters. Subsequently, each feature's weights were changed to capture the Subject and Corpus in a higher vector space, while the 'From' and 'To' feature vectors stayed unchanged and the Date vector space was lowered to the lowest priority.

C. Priority Token Generator

The software then calculates the priority tokens for the single email by comparing it to the corpus of emails using 2 methods:

1) Term Frequency-Inverse Document Frequency (TF-IDF).
2) Dictionary.

Both methods use the same stop words list for removing stop words, and the same method of removing email metadata. The program reads the tokenized emails from the output folders and calculates priority tokens for both Method 1 and Method 2, and then it outputs the tokens to output/priority.

1) Term Frequency-Inverse Document Frequency (TF-IDF):

Priority tokens using the TF-IDF (Term Frequency- Inverse Document Frequency) algorithm. A single email and a corpus of emails are taken as input, the program iterates through each token in the single email and calculates its TF-IDF score and filters out tokens with scores of zero or infinity. The top 15 tokens with the highest TF-IDF scores are then sorted and written into a text file. The TF-IDF score for a term is determined by its frequency in the single document (TF) multiplied by the logarithm of the total number of documents divided by the number of documents containing the term (IDF).

The "term frequency" (TF) component calculates how often a term appears in a document. It's computed by dividing the number of times a term occurs in a document by the total number of terms in that document. TF emphasizes terms that occur frequently within a specific document. The "inverse document frequency" (IDF) component measures the rarity of a term across the corpus. It's calculated by taking the logarithm of the ratio of the total number of documents to the number of documents containing the term. IDF highlights terms that are rare across the entire corpus but common in a specific document.

Multiplying TF by IDF results in the TF-IDF score. This score identifies terms that are both frequent in the document and unique to it, thus distinguishing them from terms that are common across many documents.

2) The Dictionary Method:

In this method, the program first populates a token dictionary by counting the occurrences of each token in a corpus of emails. Then, the frequency of each token in the dictionary relative to the total number of tokens in the corpus is calculated. Next, for a given single email, it checks if each token in the email exists in the dictionary. If found, it adds the token along with its frequency to a map representing the single email with token frequencies.
The tokens in the single-email map are sorted based on their frequencies, and the top 15 tokens are selected as priority tokens. These priority tokens, along with their frequencies, are written to a text file.

Overall, the method identifies important tokens in a single email by comparing them against a dictionary constructed from a corpus of emails, providing a means to prioritize tokens based on their frequency in the corpus.

D. Mail Response Automation

The proposed research implements an automated email response system utilizing the Simple Mail Transfer Protocol (SMTP) to promptly reply to incoming emails. This system aims to streamline communication processes by automatically generating and sending responses to incoming emails, thereby reducing response time and enhancing user experience. By leveraging SMTP, a widely adopted communication protocol for sending and receiving emails over the internet, our system ensures reliable and efficient delivery of automated replies. Through this approach, we seek to enhance productivity and efficiency in managing email correspondence, ultimately improving overall communication workflows.

This software is provided as an API service by hosting the model and can be seamlessly integrated with Gmail service.

V. RESULTS

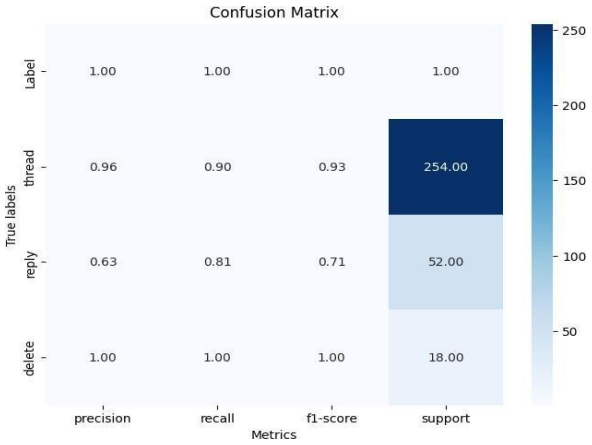

Fig. 2. Doc2vec Sentence embedding with SVM-RBF kernel and C – 1

The test result for the Doc2Vec model using the Support vector machine - Radial Basis kernel for the Hyperparameter C = 1 is displayed in Figure 5. The obtained accuracy and precision were 89.23% and 89.80%, respectively. Additionally, the metrics and the classification report are recorded. The model is performing well overall, as there are many emails correctly classified on the diagonal. However, there are some misclassifications, as shown by the non-zero values in the off-diagonal cells. The model is more likely to misclassify spam emails as not spam than vice versa. This could be because there may be more non-spam emails in the dataset than spam emails.

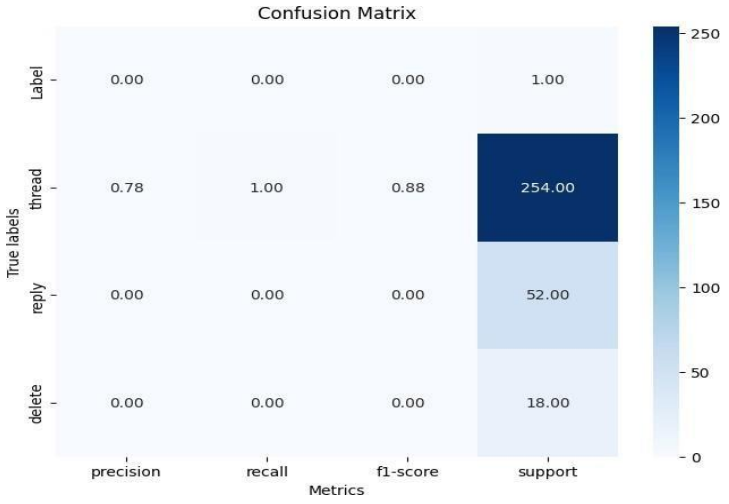

Fig. 3. Tf-Idf model with Naive Bayes classifier

The test result for the Tf-Idf Model using the Naive Bayes Classifier is displayed in Figure 3. Accuracy and precision were 78.15% and 68.57%, respectively, and the remaining labels are shown by class. Accuracy and precision were nearly zero for the removed label because the train sample size was much smaller than for the other

two labels.

The recommendation system was trained on a supervised classifier which used word embeddings, this doesn't take named entity recognition into consideration which is an useful information while processing email data.

VI. CONCLUSION

The digital age has revolutionized communication by adopting email, which is now used for both personal and professional work. However nowadays using email can be challenging due to overload of information where people get hundreds of emails daily and reading them and answering each can be a tedious task. The traditional email system doesn't really address all the problems or complexities modern communication has to face, this results in a significant loss of both time and productivity for organizations and individuals. The Smart Email Prioritization and Automatic Reply Generation System solves most of the modern complexities or problems faced, trained on Enron dataset and using TF-IDF and Doc2Vec algorithms this system can prioritize emails based on their importance and generate Reponses that context- aware. Integrating the system with existing clients for emails such as Gmail, makes it possible revolutionize the way organizations and individuals handle email communication. In conclusion, the intelligent email management system outlined in this paper represents a significant advancement in addressing the challenges posed by email overload. By harnessing the capabilities of machine learning, the system empowers users to regain control of their email inboxes, heralding a future where email communication is characterized by efficiency, effectiveness, and user empowerment.

REFERENCES

[1] A. Al-Alwani, "Improving Email Response in an Email Management System Using Natural Language Processing Based Probabilistic Methods," Journal of Computer Science, vol. 11, pp. 109-119, 2015. DOI: 10.3844/jcssp.2015.109.119.

[2] E. Horvitz, A. Jacobs, and D. Hovel, "Attention-Sensitive Alerting," CoRR, abs/1301.6707, 2013.

[3] Q. M. Ha, Q. A. Tran, and T. T. Luyen, "Personalized Email Recommender System Based on User Actions," in Simulated Evolution and Learning, L. T. Bui, Y. S. Ong, N. X. Hoai, H. Ishibuchi, and P. N. Suganthan, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 280–289.

[4] T. Hasegawa and H. Ohara, "Automatic Priority Assignment to E-mail Messages Based on Information Extraction and User's Action History," in Intelligent Problem Solving. Methodologies and Approaches, R. Logananthara, G. Palm, and M. Ali, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 573–582.

[5] S. Yoo, "Machine Learning Methods for Personalized Email Prioritization," 2010.

[6] A. Kannan et al., "Smart Reply: Automated Response Suggestion for Email," CoRR, abs/1606.04870, 2016.

[7] M. Patel, A. Shukla, R. Porwal, and R. Kotecha, "Customized Automated Email Response Bot Using Machine Learning and Robotic Process Automation," SSRN Electronic Journal, 2019. DOI: 10.2139/ssrn.3370225.

[8] M. A. Naeem, I. W. S. Linggawa, A. A. Mughal, C. Lutteroth, and G. Weber, "A Smart Email Client Prototype for Effective Reuse of Past Replies," IEEE Access, vol. 6, pp. 69453-69471, 2018. DOI: 10.1109/ACCESS.2018.2878523.

[9] N. Kumar, S. Sonowal, and S. Nishant, "Email Spam Detection Using Machine Learning Algorithms," pp. 108-113, 2020. DOI: 10.1109/ICIRCA48905.2020.9183098.

[10] S. Sinha, I. Ghosh, and S. Satapathy, "A Study for ANN Model for Spam Classification," 2021. DOI: 10.1007/978-981-15-5679-1_31.