

Sortitude – AI Email Sorter

Karan Sanghvi¹, Taher Ariwala², Vivaash Jain³, Pratham Dhruv⁴ and Mansi Kambli⁵

¹⁻⁵Department of Computer Engineering, KJ Somaiya School of Engineering, Somaiya Vidyavihar University, Mumbai, India
Email: karan.sanghvi@somaiya.edu, vivaash.jain@somaiya.edu, taher.ariwala@somaiya.edu, pratham.dhruv@somaiya.edu, prasadinipadwal@somaiya.edu

Abstract— Handling Emails efficiently is becoming a difficult task as important messages are often left below the trunk of less important emails. This problem is present among individuals who regularly receive emails, such as a student who may miss certain updates regarding their exams or assignments.

Our research identified two primary concerns about email management: the big mass of emails, which makes it difficult to notice a few important emails, and the tendency of essential messages to become lost in the heap of messages. To gain insights, we held a survey that presented user's email habit and their difficulties with the organization.

Notably 34.5% of user's make use of two email accounts, and 25.9% have three or more accounts. A major portion of user's (81%) reported frequently missing important mails due to a flood of emails, with 70.7% of people liking for categorization. To label these needs, we put forward a Chrome Extension that effectively organizes emails.

This tool will help users to search for specific emails using prompts, it also provides AI-Categorization into folders. In addition to this, a summarization feature which gives concise and precise previews. This extension aims to smooth email management and help users order essential messages.

Index Terms— Email management, organization, AI-driven filtering, Chrome extension, Prompt- Based Search, Email Categorization, Summarization, Productivity Tools.

I. INTRODUCTION

Nowadays, managing emails has become a difficult challenge for many people as important emails often get lost in a flood of less significant emails.

For example, consider that a user has subscribed to services like Swiggy to purchase food or a student who receives a frequent amount of educational mail from their school/college. In such cases, critical emails like assignment deadlines can be easily missed by the student.

This problem includes two main issues: because of the overwhelming volume of emails, it is difficult to track the important ones, and important mails get buried in the clutter of covering the less important mails. To better understand the challenges faced by individuals, we surveyed them to gather insights on the problems they faced in managing emails and discussing potential solutions.

To know about the number of email accounts people actively use, 34.5 percent of respondents reported using two accounts, while 25.9 percent of respondents use three or more accounts. With this, 29.3 percent of users often miss emails, whereas 51.7 percent sometimes miss important emails due to an overload of emails. Interestingly, 70.7 percent of the users want their emails to be categorized into different folders which include: Finance, Education, Due/Pending, and much more.

Furthermore, 91.4 percent of users wish that they could see a summary of their emails by simply hovering over the message, and the same percentage showed an interest in using an AI prompt feature which would help them locate their emails. Based on the responses of the users, we have proposed a solution to develop a Chrome extension that will solve the problems of email management for users. This Chrome extension allows users to type custom prompts by describing what type of mail they wish to see, and it will display the relevant mail in real-time. With this, the extension will automatically categorize emails into different folders such as Finance, Education, Due/Pending, Newsletters, Social/Networking, and Updates/Subscriptions. Additionally, it offers a summarization feature that provides a concise two-line summary of each mail using the AI prompt feature available. This feature will enable users to see the mail in a concise format without opening and reading the whole message.

By simplifying all these features, this Chrome extension transforms the way users manage their inboxes by offering an organized, efficient, and user-friendly experience.

II. LITERATURE SURVEY

For the implementation of our project, we conducted a deep review of relevant research papers on similar applications. This guided us to learn about the features, challenges, and innovations in existing solutions which gave a great approach to our project. Below are the research papers we reviewed:

This paper puts forward query-oriented text summarization that extracts sentences dependent on 11 important features. The result is validated by ROUGE metric. [1] The paper develops an email manager which understands, organizes, and manages information in emails by semantic content relations. Current tools lack this, making it challenging for people to manage. [2] This paper displays the results of many supervised machine learning models for classifying Turkish news text in 5 categories, with Multinomial Naive Bayes having the best accuracy (90 percent). Paper suggests that Naive bayes is a good classifier. [3] This paper shows machine learning algorithms for recommenders system, on user-based and content based filtering, including all their perks and cons, and evaluation metrics. With Movielens data it implements these algorithms, presenting recommendations. [4] This paper marks email classification to reduce overhead by categorizing emails on content topics. A suggested CNN-BiLSTM model outperforms standalone CNN and BiLSTM models, achieving better accuracy. [5] This paper shows a forensic method using FP-growth algorithm to identify writing patterns in emails. The AHP method calculates pattern weights to improve forensic analysis. [6] This paper traverses machine learning algorithm for text summarization by extracting key phrases and queries. A study if supervised and unsupervised method is conducted, showing an automated agent prototype to improve electronic media support. [7] This research puts forward an email spam classification using Naive Bayes and support Vector machine to categorize spam email on content.

These use F-measure, recall, precision to determine the better of both. [8] This study investigates agile engineering techniques applied to Korean-based large languages (LLMs) using the Question Transformation Module (QTM) to enhance conversational response generation The proposed QTM, which reformulates prompts into three distinct question methods , the's SKT GPT-2 and Kakaobrain With models like KoGPT-3, it provides a performance improvement of 11.46 percent in unmodified queries. [9] This paper examines the performance of MySQL, Cassandra, and HBase in implementing heavy-duty authoring services for web-based REST applications, with specific performance requirements aimed at identifying focused flexibility increasing trend towards NoSQL databases due to their scalability and support for structured and unstructured data and to help users choose the most appropriate NoSQL solution on a basis. [10]

After a comprehensive search of relevant research papers, we developed the structural design of the chrome extension. This diagram depicted in Figure 1, highlights the basic functionality and design features of the extension, which provides a clear description of its intended capabilities and user interface.

III. SYSTEM ARCHITECTURE

Figure 1 shows a detailed project layout, showing how the Chrome extension integrates with the front-end and back-end functionality and interacts with the user. This provides a general overview of the extension's workflow, focusing on user interaction between actions, front-end interactions, and back-end processes.

IV. IMPLEMENTATION

This section focuses on the detailed implementation of the Chrome Extension. It discusses the functionality of each feature with their implementation process along with the dependencies used to implement each feature.

A. Chrome Extension

To access the Gmail data of the user for our chrome extension, we created a project on Google Cloud Console. This allows Google to monitor and track the activities to prevent any misuse. The steps involved in signing to the Google Cloud Console includes:

- Accessing the project dashboard
- Creating a new project
- Filling in the project details
- Enabling the Gmail API

After creating the project, the OAuth client is prepared to define the scopes of access to the user's account. Once this process is completed, a `credentials.json` file is generated which is integrated into the projects source code for sending requests to the Gmail API. After the above process of Google Cloud Console is completed, we set up the process of the Chrome Extension. The flowchart depicted in (Figure 1) shows the process of the Chrome Extension. The process begins when the user downloads the extension from the chrome webstore. The extension first checks if the user is new. If they are, then a welcome page will be displayed. If the user has returned to the extension, the system will check if the refresh token used for authentication has expired. If the token has expired, then it verifies the status of the access token and refreshes it if needed. Once the tokens are valid, the extension retrieves and displays the services of the extension to the user. It checks if there is a previous entry in the database, if so, it extracts only new emails; or else it extracts the last 500 emails and stores them in a excel format. If the user is not authorized, the system requests authorization and continues based on the user's status.

To implement the Chrome Extension, below are the languages used:

- ReactJS: used for building the interface of the extension.
- ExpressJS: for handling the server side logic and manage API requests.
- NodeJS: for building the backend of the application to handle the requests from the frontend.
- MySQL: to store the user information in a structured manner.
- CSS and Tailwind CSS: these styles used to provide a user -friendly experience to the user.
- Sequelize.js: is an Object Relational Mapping library to simplify database tables to objects.

B. Categorization Model

The main purpose of the segmentation model is to categorize emails into folders based on some user-defined fields such as: Finance/Banking, Education, Outstanding/Pending Newsletter, Social Media, Updates/Subscriptions Displays specific groups needed and each to assist in the labeling process to understand specific email content relevant to learning. The steps to be implemented are:

Step 1: Data Collection

Data sets of emails with different contents have been collected.

Step 2: Pre-processing Data

- Data Reduction:
 - Removed HTML tags to remove readable text
 - All text has been converted to lowercase to maintain consistency in the text reducing unnecessary changes.
 - Eliminated special characters to highlight textual information.
 - Eliminated generic, wordless words that have no special meaning.
- Text Tokenization: Separated content into searchable parts and break each email into words or tokens.
- Lemmatization: Got the actual meaning of the word is considered as it becomes more appropriate for certain types of emails where the context is important.
- Feature Labelling: Manually assigned categories to different subsets of the collected data.

Step 3: Data Analysis

- Text Analysis:
 - The distribution of sessions across groups was examined to ensure equal distribution of data.
 - Identify terms commonly associated with each category to better understand key symbols.
 - The average length of emails was examined by group, as some emails may have short text.
- Feature Removal:
 - Bag Of Words: Rotates the text numerically to show frequency of words.
 - TD-IDF: It was used to check the meaning of words in a data structure.

- Word Embeddings: captured the relationships between words by depicting each word in a vector space.

Step 4: Model Selection and Training

- Naïve Bayes: selected this model to gain the initial performance metrics. It also helps to identify the areas of improvement which ensures the dataset is structured.
- Model Training: converted the dataset into a training set to test the performance. With this we assessed its accuracy, precision, recall and F1 score on the test set.

Step 5: Model Evaluation and Refinement

- Hyperparameter Tuning:
 - Adjusted the parameters like learning rate to increase the performance of the model.
 - Used cross validation technique to test different configurations and select the best configuration.
- Feature Engineering: experimented the dataset with different features to improve accuracy.
- Model Optimization: adjusted the model settings to enhance efficiency, speed and memory usage by balancing on accuracy with low cost.

To perform the implementation of the above steps below are the libraries which we used:

- Pandas: used for the data processing and manipulating structured data in the form of DataFrames.
- NumPy: used to support the multi-dimensional arrays for data preprocessing along with some mathematical functions.
- SpaCy: used for advanced tokenization and lemmatization to improve its speed and efficiency for large volumes of datasets.
- Scikit-Learn: used for feature extraction like Bag Of Words and TF-IDF and also as a machine learning framework.
- Regex: used for pattern matching, removing special characters and finding particular sequences in text.
- Gensim: used to generate word embeddings.
- Matplotlib: used for data visualization like plotting word frequency and categorization of emails.
- Joblib: used to save and load the training models and vectorizers.

C. Summarization Model

The main purpose of the summarization model to give the summary of each email in maximum two lines. This feature of the Chrome Extension helps the user to read lengthy emails in a short span of time. The steps implemented are:

Step 1: Imported Libraries and Model

To implement the summarization model, below are the libraries which we used:

- Pandas: is a python library used to load, manipulate and save the dataset.
- Transformers: is a python library which provides pre-trained models for summarization.

The pre-trained model used for summarization is the Bidirectional And Auto-Regressive Transformer. It is also called as BART model which is loaded with a tokenizer used to prepare the email text for processing. It is a pre-trained model which makes it suitable to generate summaries of textual content.

Step 2: Summarization Function

- This function is used to convert into tokens for the BART model to understand.
- Afterwards, it uses the pre-trained model's text generating capabilities to give a summary by parameters controlling the length and structure.
- The output is decoded into human readable text.

Step 3: Generating Summaries for Each Email

The program iterates through each email in the dataset. For each email:

- If the body is valid, the summarization function generates a summary in two lines.
- If the email has no body or invalid, a default message is seen saying: "No valid summary" which indicates that the email cannot be summarized.

To perform the implementation of the above steps, below are the libraries and model which we used:

- Pandas: a python library used for data manipulation. With this it helped to save and load the excel file to save the data.
- Transformer: a library for natural language processing tasks. This library provides some pre-trained models and tokens.

- Bidirectional And Auto-Regressive Transformer (BART): it contains; BartForConditionalGeneration which is a pretrained model from Facebook used for summarization. And BartTokenizer which takes the input text into a format BART model can process.
- facebook/bart-large-cnn: this is pre-trained model which is used for summarization. It has 12 layers which used the CNN/DailyMail dataset for summarization tasks which makes it best for processing and summarizing text.

D. AI Prompt

The main purpose of the AI Prompt feature is to integrate the summarization and categorization model. In this feature, according to the prompts made by the user, they receive customized responses tailored to specific context. Below are the steps performed to implement the prompt feature.

Step 1: User Interaction

- Ask user to enter name:
 - When the user initiates the AI prompt, the AI first asks for the user to enter their name.
 - This is a first step to validate the interface and can be used later in the session for the user's specific configuration or context.

After receiving the user's name, the AI offers two distinct services to the user:

- Summarization
- Categorization

Step 2: Service Selection

- Summarization Service: If the user selects the summary service, the AI can insert specific prompts to summarize that information, especially email summaries. Some provocative examples are:
 - "I'm summarizing all my emails from last week."
 - "Summarize emails for today."

Such prompts the user can enter to read bulk emails in a easy manner.

- 2) Categorization Service: When the user selects categorization, the AI has the ability to categorize emails into pre-defined folders according to user prompt. Some provocative examples are:
 - "Consolidate my finance emails into finance/banking folders."
 - "Give me the last few days for this week's projects."

These prompts allow users to filter and categorize content based on content, topics, or timelines.

Step 3: Activity And Response

- Backend Integration With Models: After receiving the user's selected prompt, the backend invokes the summarizing or classifying model accordingly. For categorization, according to the keywords and subject of the email, the backed segregates the emails into specific folders. Whereas, the summarization converts the text into tokens, summarizes it and gives the summary in two lines.
- Output Generation: The extension generates and displays responses to the user according to their prompts. It incorporates summary of emails and categorizing of mails into different folders, allowing users to quickly find the information they want in a user-friendly format.s

Step 4: User Interaction

- Prompt Flexibility: The AI prompt feature gives the ability to the user to write as many prompts or switch to any other service they wish to.
- Embedded Growth: The AI feature is able to handle multiple presentations in a single session, by creating a more dynamic and interactive experience to the user. This format provides a more efficient and flexible workflow to the user.

To implement the AI prompt feature, below are the languages and libraries used.

- ReactJS: For the AI prompt feature, React will maintain the basic layout of the chatbot interface, displaying the chat window and any input/output elements as part of the AI prompt feature.
- react-chatbot-kit: This kit allows you to create a chat interface where users can interact with the AI. It provides tools for configuring various chatbot features, such as message analytics, response generation, and conversation flow management. It's highly customizable, so you can combine prompts, responses, and API calls for the AI backend.

In conclusion with this section, we saw the steps implemented for each feature of the chrome extension. With the steps, we saw the python libraries and languages used to implement these features.

V. RESULTS

This section focuses on the results obtained when we integrated the summarization and categorization model in the AI prompt feature, along with the implementation of the Chrome Extension. When the user opens the Chrome Extension, the user can see the welcome screen as depicted in Figure 2. Afterwards, when the user clicks the next button, they will be able to see the functionalities provided by the extension as depicted in Figure 3. After seeing the functionalities, the user clicks the authorize button to get certain rights to perform the features as shown in Figure 4. After the user authorizes, the extension starts fetching the emails as seen in Figure 5. After the emails are fetched, they get a confirmation window that the emails are processed and they can close the window if they wish to as seen in Figure 6. In Figure 7, they can see how many emails are accessed and after some stipulated time the user can use the services. After the stipulated time is over, the user can access the AI prompt feature. In Figure 8, we can see the full implementation of the AI prompt feature along with the different prompts entered by the user.

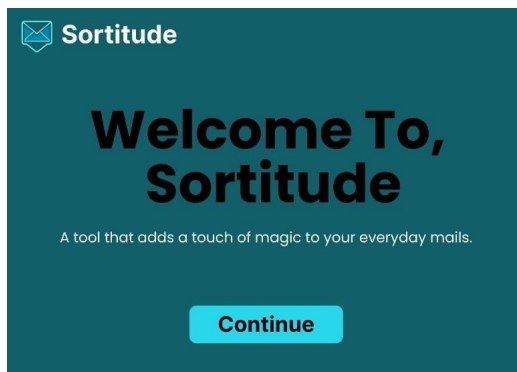


Fig 2: Welcome Screen

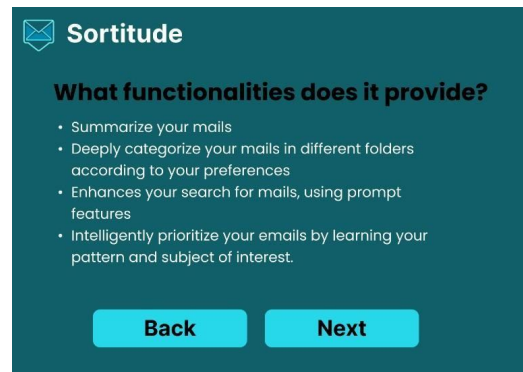


Fig 3: Functionalities

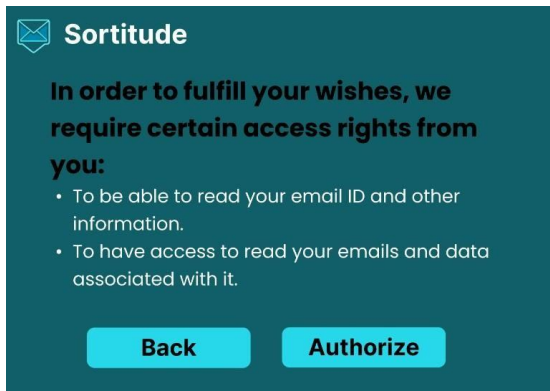


Fig 4: Information To Access Rights

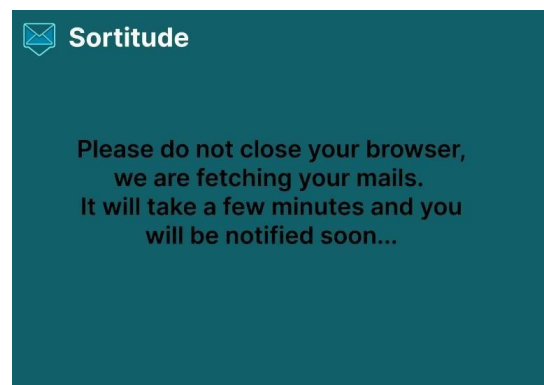


Fig 5: Fetching Emails

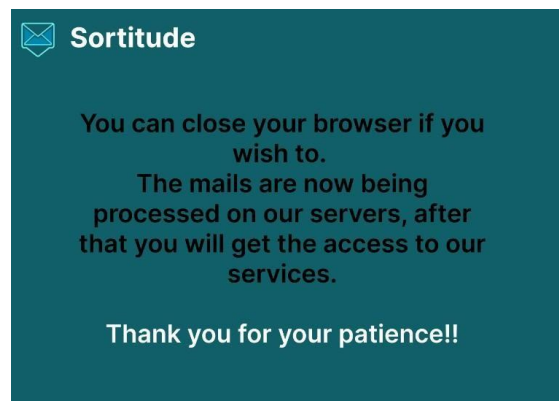


Fig 6: Confirmation Message Of Fetching Emails

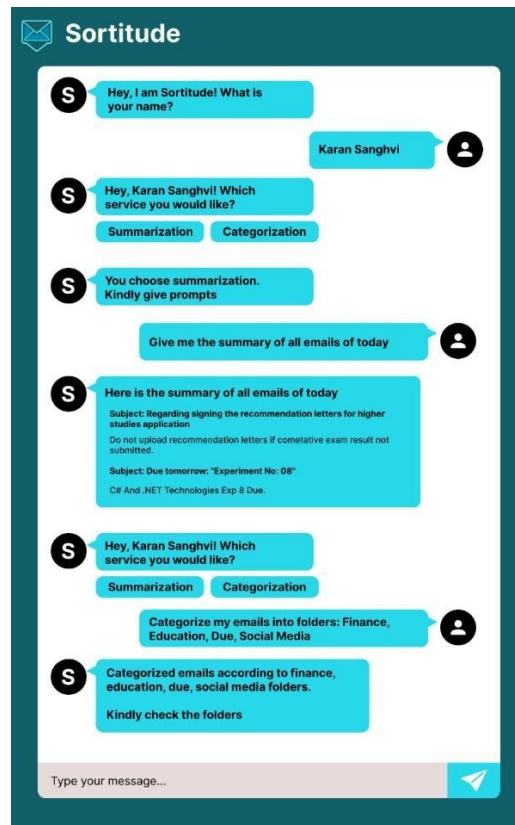


Fig 7: AI Prompt Feature

VI. CONCLUSION

Email management is very important, yet quite challenging. Our research underlines the need of organization and prioritization within the email system, as many find it difficult to track essential messages among a flood of less important messages. Through surveys we pointed out common struggle points, such as being vulnerable to miss critical emails and the desire for a better categorization and summarization. To solve these needs, we implemented a Chrome Extension that enables users to search for specific types of mails via prompts and automatically categorizes messages into folders such as Subscription, Education, etc. it also provides a precise preview of the emails. By integrating AI-filtering, categorization, summarization, our solutions offer a smooth, user-friendly approach to management that allows us to prioritize and locate important emails. This system represents a step towards a more organized, productive inbox, reducing time and effort managing emails and letting them stay focused on tasks. Future work may have advanced machine learning model to enhance accuracy and adaptability.

VII. FUTURE SCOPE

The field of Sortitude has made major improvements in organization and classification. However, with the rising need of enhancing tools, Sortitude can further support users by implementing features that help with deadlines, to-do list and response suggestions. Here, we highlight potential future scope to Sortitude, which streamlines workflow and improves efficiency by having emails as insights.

A. Focus on Emails with Deadlines

In any user's life, handling time sensitive tasks is an important part. An improvement in Sortitude could include a feature that establishes and shows emails containing deadline related information. This feature would use Natural Language Processing (NLP) to detect words and phrases that describe a deadline, such as 'due', 'by ..', and etc. In addition. Machine Learning models will help to find less obvious deadlines where urgency is required. By marking emails with deadlines, one can quite easily spot and order their work, reducing the chances of missing important mails and allowing efficient time management.

B. Task Extraction and To-Do List

Many Emails may contain implanted tasks, which we often have to track separately. A task extraction in a To-Do list feature will help the user by simplifying the process by automatically recognizing tasks in the mail and organizing them into one To-do List. Using Natural Language Processing, the system will parse through the mails content for certain verbs and keywords related to tasks, such as “review”, “complete”, “up next”,etc. By spawning a well-defined list of tasks from email content, users will have a clear view of their responsibilities. This To-Do list could integrate with calendar, further supporting its productivity. This feature will highlight task management, confirming that no items are missed within the inbox.

C. Response Suggestion

Replying to routine emails can become very tedious in any user's life. A response suggestion feature would aim to remove this tediousness by spawning replies based on emails content and tone. Exploiting NLP models, this feature will look on each email and suggest a appropriate response. For example, if an email contains an appointment request, the feature may suggest time slot in the reply to the email. The suggested replies could be customizable, allowing users to add their personal touch to the replies generated. This feature will help save a lot of time for general emails.

REFERENCES

- [1] M. Afsharizadeh, H. Ebrahimpour-Komleh and A. Bagheri, "Query-oriented text summarization using sentence extraction technique," 2018 4th International Conference on Web Research (ICWR), Tehran, Iran, 2018, pp. 128-132, doi: 10.1109/ICWR.2018.8387248.
- [2] T. Ayodele, G. Akmayeva and C. A. Shoniregun, "Machine learning approach towards email management," World Congress on Internet Security (WorldCIS-2012), Guelph, ON, Canada, 2012, pp. 106- 109.
- [3] F. Gürçan, "Multi-Class Classification of Turkish Texts with Machine Learning Algorithms," 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2018, pp. 1-5, doi: 10.1109/ISMSIT.2018.8567307.
- [4] A. Nawrocka, A. Kot and M. Nawrocki, "Application of machine learning in recommendation systems," 2018 19th International Carpathian Control Conference (ICCC), Szilvasvarad, Hungary, 2018, pp. 328-331, doi: 10.1109/CarpathianCC.2018.8399650.
- [5] Z. Luo and F. Zulkernine, "An Intelligent Email Classification System," 2023 IEEE Symposium Series on Computational Intelligence (SSCI), Mexico City, Mexico, 2023, pp. 1126-1131, doi: 10.1109/SSCI52147.2023.10371921.
- [6] Yiyun Zhang, Yanhua Liu and Guolong Chen, "A solution of anonymous email identification based on writing structural pattern," 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Zhangjiajie, 2015, pp. 1525-1531, doi: 10.1109/FSKD.2015.7382171.
- [7] K. Lanyo and A. Wausi, "A Comparative Study of Supervised and Unsupervised Classifiers Utilizing Extractive Text Summarization Techniques to Support Automated Customer Query Question-Answering," 2018 5th International Conference on Soft Computing And Machine Intelligence (ISCMI), Nairobi, Kenya, 2018, pp. 88-92, doi: 10.1109/ISCMI.2018.8703237.
- [8] T. M. Ma, K. YAMAMORI and A. Thida, "A Comparative Approach to Naïve Bayes Classifier and Support Vector Machine for Email Spam Classification," 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE), Kobe, Japan, 2020, pp. 324-326, doi: 10.1109/GCCE50665.2020.9291921.
- [9] D. Park, G. -t. An, C. Kamyod and C. G. Kim, "A Study on Performance Improvement of Prompt Engineering for Generative AI with a Large Language Model," in Journal of Web Engineering, vol. 22, no. 8, pp. 1187-1206, November 2023, doi: 10.13052/jwe1540-9589.2285.
- [10] V. D. Jogi and A. Sinha, "Performance evaluation of MySQL, Cassandra and HBase for heavy write operation," 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, India, 2016, pp. 586-590, doi: 10.1109/RAIT.2016.7507964.