



Smart Email Assistant: An AI-Powered System for Email Summarization, Priority Classification and Automated Reply Generation

Mrs. I. A. Jannathul Firthous^{*1}, Loghasree G S^{*2}

^{*1}Assistant Professor, Sri Shakthi Institute of Engineering and Technology, India

^{*2}Student, Sri Shakthi Institute of Engineering and Technology, India

ABSTRACT

Email remains the primary channel of professional and academic communication, yet the rising volume of messages leads to information overload, missed deadlines and a significant share of working time spent on routine correspondence. This paper presents the design of a Smart Email Assistant, an AI-powered system that connects to a user's mailbox and automatically (i) classifies incoming messages by priority and category, (ii) produces concise abstractive summaries of long messages and threads, (iii) drafts context-aware, tone-controlled replies for user approval, and (iv) extracts tasks, deadlines and meeting requests for scheduling. The system combines a modular natural language processing pipeline built on pre-trained transformer models with a human-in-the-loop dashboard, so that no message is sent without explicit user consent. A privacy-first architecture based on OAuth 2.0 authorization, encrypted storage and minimal data retention is adopted. The paper describes the architecture, the methodology of each module, the implementation stack and an evaluation plan using standard metrics such as ROUGE, F1-score and response-time reduction. The proposed assistant is expected to reduce the time users spend triaging and answering email while keeping the user in full control of outgoing communication.

Keywords: Smart Email Assistant, Natural Language Processing, Email Summarization, Priority Classification, Automated Reply Generation, Transformer Models

1. Introduction

Electronic mail is still the backbone of organisational, academic and customer communication. Professionals receive dozens to hundreds of messages every day, and the effort of reading, prioritising and answering them has been described as "email overload" since the earliest studies of personal information management (Whittaker & Sidner, 1996). Important requests are buried under newsletters, notifications and long reply chains, which results in delayed responses, missed deadlines and reduced productivity.

Conventional email clients offer rule-based filters, labels and spam detection, but these mechanisms are static and do not understand the meaning of a message. Recent advances in natural language processing (NLP), particularly pre-trained transformer models (Vaswani et al., 2017; Devlin et al., 2019; Brown et al., 2020), make it possible to understand, summarise and generate text with a quality that was previously unattainable. These capabilities can be applied to the mailbox to build an intelligent assistant that works alongside the user.

This paper proposes a Smart Email Assistant that integrates four capabilities in a single workflow: priority classification, summarization, reply generation, and task and event extraction. The key contributions are: (a) a modular, extensible architecture that connects to common mail providers; (b) a human-in-the-loop design in which all automatically generated replies are drafts that need user approval; (c) a privacy-oriented data-handling strategy; and (d) an evaluation framework that measures both model quality and real user benefit.

2. Related Work

Early research on email focused on filtering. Sahami et al. (1998) applied a Bayesian classifier to junk e-mail, and later work extended text classification to folder prediction and categorisation using public corpora such as the Enron email dataset (Klimt & Yang, 2004). Research on user behaviour showed that reply likelihood and latency depend on message content, sender relationship and context (Yang et al., 2017), which motivates content-aware prioritisation.

For text generation, Kannan et al. (2016) introduced Smart Reply, a deployed system that suggests short responses to emails using sequence-to-sequence learning. Since then, encoder-decoder and large language models such as BART (Lewis et al., 2020), T5 (Raffel et al., 2020) and PEGASUS (Zhang et

al., 2020) have set strong baselines for abstractive summarization, while large autoregressive models (Brown et al., 2020) can write fluent, context-aware replies from a few examples.

Existing solutions typically address one task in isolation, for example only spam filtering or only short canned replies. Table 1 summarises the gap that the proposed assistant aims to close.

Table 1 - Comparison of email-handling approaches.

Feature	Rule-based filters	Smart Reply	Proposed assistant
Spam / priority detection	Yes (static)	No	Yes (learned)
Thread summarization	No	No	Yes
Full reply drafting	No	Short only	Yes, with tone
Task / event extraction	No	No	Yes
User approval before send	—	Yes	Yes

3. Problem Statement and Objectives

Users lack a lightweight tool that reduces the time spent reading and answering email without removing their control over what is sent. The main objectives of this work are:

- to automatically rank incoming emails by urgency and importance;
- to generate short, accurate summaries of long emails and threads;
- to draft professional replies in a tone selected by the user (formal, friendly, concise);
- to detect actionable items, deadlines and meeting requests and convert them into tasks or calendar entries; and
- to protect user privacy through secure authorisation and minimal data retention.

4. Proposed System

4.1 System architecture

The Smart Email Assistant follows a layered architecture, shown in Fig. 1. A fetcher module retrieves new messages from the mail server through the provider API or IMAP. After preprocessing, each message is passed to four NLP modules that operate independently, which keeps the system modular and allows individual models to be replaced. Their outputs are presented in a dashboard where the user can review, edit, approve or schedule actions. Corrections made by the user are logged as feedback to improve later predictions.

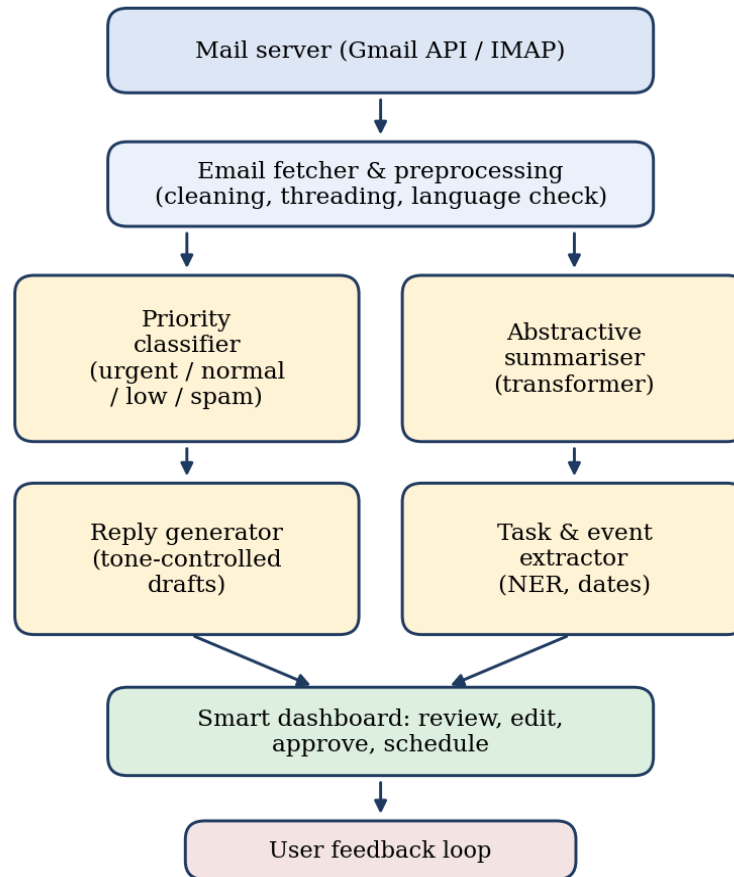


Fig. 1 - Architecture of the Smart Email Assistant.

4.2 Email ingestion and preprocessing

Access to the mailbox is granted by the user through OAuth 2.0, so that the system never stores the account password. Incoming messages are parsed into sender, recipients, subject, body, timestamp and thread identifier. Preprocessing removes HTML markup, signatures, disclaimers and quoted text from earlier replies, normalises whitespace and detects the language. Messages in a thread are grouped so that the summariser and reply generator receive the full conversational context.

4.3 Priority classification

Each email is assigned a category (urgent, normal, low priority or spam/promotional). The classifier uses text features from a fine-tuned transformer encoder together with metadata such as sender relationship, previous reply history, presence of deadlines and the number of recipients. A priority score is then computed as a weighted combination of these signals:

$$S = w_1U + w_2I + w_3D + w_4K \quad (1)$$

where U is the urgency probability predicted by the classifier, I the sender importance, D the deadline proximity, K the presence of user-defined keywords, and w_1 – w_4 are weights that are tuned from user feedback. Emails are displayed in descending order of S .

4.4 Summarization

Long emails and threads are condensed by an abstractive summariser based on a pre-trained encoder–decoder model such as BART or T5. For very long threads, the text is split into chunks, each chunk is summarised, and the partial summaries are merged in a second pass. The output consists of a one-line gist and, optionally, three to five bullet points of key information. Extractive highlighting of names, dates and amounts is used to reduce the risk of the summary omitting critical facts.

4.5 Reply generation

For messages that need a response, the generator produces up to three draft replies. The prompt contains the thread context, the user's selected tone and length, and optional user instructions such as "accept the meeting but propose Friday". A few examples of the user's earlier sent emails may be added to match their writing style. The system never sends a reply automatically; the user must review and approve each draft, and unsupported claims, commitments or attachments are flagged for attention.

4.6 Task and event extraction

Named entity recognition and date-expression parsing identify action items, deadlines, locations and meeting proposals. Detected items are shown as suggestions that can be added to a to-do list or calendar with one click, and reminders are scheduled before the deadline. Scheduling conflicts with existing calendar events are highlighted.

4.7 Privacy and security

Because email contains sensitive information, the design follows the principle of data minimisation. Access tokens are stored encrypted, all communication uses TLS, message bodies are processed in memory and only derived metadata (labels, summaries) is retained for a limited period that the user can configure. Users can revoke access and delete all stored data at any time. Where possible, models are run locally or on a trusted server rather than sending content to third parties.

5. Implementation

The prototype is designed with a Python back end exposing REST endpoints and a web-based front end. Table 2 lists the main components and the technologies used for each.

Table 2 - Components and technologies.

Component	Technology
Mail access	Gmail API / IMAP, OAuth 2.0
Back end	Python, FastAPI
NLP models	Hugging Face Transformers (BERT, BART/T5), spaCy
Storage	PostgreSQL (encrypted), Redis queue
Front end	React web dashboard
Deployment	Docker containers, HTTPS

Incoming mail is processed asynchronously through a task queue so that the dashboard stays responsive. Models are fine-tuned on public email corpora and then adapted with user feedback.

6. Evaluation Methodology

The assistant is evaluated at two levels. At the model level, the priority classifier is measured using precision, recall and F1-score on a labelled email set, the summariser using ROUGE scores (Lin, 2004) together with human ratings of faithfulness and readability, and the reply generator using human ratings of relevance, tone and correctness, supplemented by the proportion of drafts that users send with little or no editing. At the system level, a user study compares the time taken to process a fixed set of emails with and without the assistant, and records user satisfaction through a questionnaire. Table 3 lists the metrics.

Table 3 - Evaluation metrics.

Module	Metric
Priority classifier	Precision, recall, F1-score
Summarizer	ROUGE-1/2/L, faithfulness rating
Reply generator	Relevance, tone, edit rate
Overall system	Time saved, user satisfaction

7. Discussion

The expected benefits are a shorter time to triage the inbox, fewer overlooked urgent messages and faster, more consistent replies. The human-in-the-loop design limits the main risks of generative models. Nevertheless, several limitations must be acknowledged. Abstractive summarisers may omit or distort details; generated replies may sound generic or misjudge social context; and priority models may reflect biases present in training data. Handling confidential email also requires strict compliance with applicable data-protection regulations. These issues are mitigated through approval steps, highlighting of key facts, transparent display of why an email was ranked as urgent, and strong privacy controls.

8. Conclusion and Future Work

This paper presented the design of a Smart Email Assistant that unifies priority classification, summarization, reply drafting and task extraction in a privacy-conscious, user-controlled workflow. By building on pre-trained transformer models and keeping the user in the loop, the system aims to reduce email overload without sacrificing trust. Future work includes full implementation and a user study with the metrics described above, multilingual support, personalisation of writing style through on-device learning, voice-based interaction, integration with team chat and project-management tools, and attachment understanding.

Acknowledgements

The authors thank the Department and the Institution for the guidance and facilities provided during this work.

References

- Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT*, 4171–4186.
- Kannan, A., Kurach, K., Ravi, S., et al. (2016). Smart Reply: Automated response suggestion for email. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 955–964.
- Klimt, B., & Yang, Y. (2004). The Enron corpus: A new dataset for email classification research. *Machine Learning: ECML 2004, LNCS 3201*, 217–226.
- Lewis, M., Liu, Y., Goyal, N., et al. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Proceedings of ACL*, 7871–7880.
- Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 74–81.
- Raffel, C., Shazeer, N., Roberts, A., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1–67.
- Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. *AAAI-98 Workshop on Learning for Text Categorization*.
- Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998–6008.
- Whittaker, S., & Sidner, C. (1996). Email overload: Exploring personal information management of email. *Proceedings of ACM CHI 1996*, 276–283.
- Yang, L., Dumais, S. T., Bennett, P. N., & Awadallah, A. H. (2017). Characterizing and predicting enterprise email reply behavior. *Proceedings of ACM SIGIR*, 235–244.
- Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. *Proceedings of ICML*, 11328–11339.