iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** V **Month of publication:** May 2025

DOI: https://doi.org/10.22214/ijraset.2025.71297

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

Asthra MailGuard: A Privacy-First Hybrid AI Email Assistant Using ML and Local LLMs

Mynamapti Sri Ranganadha Avinash[1], Dr. Ruhin Kouser R[2]
Department of Computer Science and Engineering, Presidency University, Bangalore

Abstract: ***In the era of digital communication, the volume and complexity of email traffic continue to rise, leading to challenges in managing, filtering, and responding effectively to diverse messages. While traditional spam filters offer basic classification, they lack personalization, contextual understanding, and offline privacy guarantees. This paper presents Asthra MailGuard, a privacy-first, AI-powered command-line email assistant that classifies, filters, and responds to emails using a hybrid architecture—combining a lightweight Logistic Regression model with a fallback to Hermes 3, a powerful local Large Language Model (LLM) via Ollama.***

The system introduces multiple innovations including a Safe Mode for privacy enforcement, domain-based rules, a self-learning feedback loop, and personalized response drafting guided by user tone. With support for offline execution, Asthra MailGuard ensures total data control while offering intelligent predictions and response generation. Evaluated on real-world email samples, it achieves improved accuracy and usability, highlighting its potential as a reliable, scalable solution for email management in both personal and professional environments.

Keywords: Email Classification, Machine Learning model, Hybrid AI, Privacy-Preserving NLP, Large Language Models, Ollama, Self-Learning Systems, Hermes 3, Logistic Regression, Offline Assistant.

I. INTRODUCTION

Email remains a cornerstone of digital communication across personal, academic, and professional contexts. Despite the evolution of messaging platforms, the volume and complexity of email correspondence continue to grow. This has led to a surge in unwanted emails, such as spam, advertisements, and promotional content, alongside important messages like job alerts, result notifications, or financial updates. While traditional spam filters offer basic classification, they are often rule-based and static, failing to adapt to the unique communication patterns and evolving needs of users. Additionally, existing solutions often rely on cloud infrastructure, raising serious concerns about data privacy, especially when sensitive content is analyzed or stored remotely. Modern users now seek intelligent systems that are not only effective but also privacy-conscious and adaptable. In this context, there is a pressing need for a system that can classify emails accurately, adapt to user feedback, and operate entirely offline. Asthra MailGuard addresses this challenge by integrating Machine Learning and Large Language Models (LLMs) in a hybrid setup. It combines a lightweight logistic regression model for fast predictions with an advanced fallback mechanism leveraging Hermes 3 (LLM via Ollama) for low-confidence cases. The system supports domain-specific rules, retraining, safe mode restrictions, and personalized reply drafting. This introduction sets the stage for a robust, privacy-first AI email assistant with real-world usability and scalability.

II. LITERATURE REVIEW

Over the past two decades, numerous techniques have been developed to tackle the problem of email classification. Traditional models like Naive Bayes, Support Vector Machines (SVM), and Logistic Regression have been used to separate spam from legitimate messages based on textual features and probabilistic thresholds. These systems are lightweight and suitable for real-time filtering, but they suffer from limitations in adaptability and contextual understanding.

With the rise of deep learning, Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and Transformer architectures have demonstrated significant improvements in text classification. However, such models are data-hungry, compute-intensive, and usually reliant on cloud-based services like Google's Gmail categorization or Microsoft's Outlook filtering. These services introduce privacy trade-offs, especially for users dealing with sensitive content.

Recently, Large Language Models (LLMs) such as GPT-3, GPT-4, and Hermes 3 have proven effective at understanding context, semantics, and tone in natural language. LLMs outperform traditional models in handling ambiguous or novel content, but they come with the burden of high computational cost and data security risks when accessed via API.

©IJRASET: All Rights are Reserved

Despite the technical advancements, most existing systems lack key capabilities like personalization, offline adaptability, or privacy-first design. This paper builds upon these foundations to introduce a system that blends lightweight ML with LLM intelligence, while preserving user control and local execution.

III. RESEARCH GAP & OBJECTIVES

Despite the evolution of email filtering technologies, most existing solutions still fall into two categories — rule-based filters or cloud-powered AI models. Rule-based filters, while fast, are rigid and cannot adapt to new patterns or user-specific behaviors. On the other hand, cloud-based AI systems such as Gmail's or Outlook's categorization engines offer better intelligence but compromise on privacy, personalization, and offline usability. They also provide little to no transparency or retraining options for end users. Even among recent advancements using Large Language Models (LLMs), solutions are either API-based (like OpenAI or Bard) or tightly coupled with proprietary platforms, making them inaccessible for offline, customizable, or secure deployments. Most academic approaches focus on improving classification accuracy but overlook real-world user needs like context-aware replies, domain-based overrides, or confidence fallback mechanisms.

A. Asthra MailGuard

Addresses this research gap by integrating:

- A lightweight, offline Logistic Regression classifier for real-time inference
- An LLM fallback system (Hermes 3 via Ollama) for low-confidence emails
- A Safe Mode to block sensitive content from LLMs
- A feedback loop and retraining pipeline for self-learning
- Draft response generation based on user tone and email context

B. Objectives

- To design a privacy-first, offline email classification system using hybrid AI (ML + LLMs)
- To implement safe and intelligent fallback using local LLM (Hermes 3) for ambiguous emails
- To enable user feedback-based retraining for personalization
- To generate tone-aware, professional email replies directly within CLI

IV. PROPOSED METHODOLOGY

Asthra MailGuard is built around a **modular hybrid AI pipeline** that leverages the speed of traditional machine learning with the contextual strength of local large language models (LLMs). The entire system runs offline and is optimized for both performance and privacy.

The methodology is divided into the following stages:

1) Email Input & Preprocessing

Emails are manually input through a CLI interface. The raw content is passed through a custom cleaning module that removes stopwords, special characters, URLs, and normalizes the text to lowercase. This prepares the content for consistent vectorization.

2) Ml Classification With Confidence Score

A Logistic Regression model, trained on labeled email data, performs the initial classification. The system also generates a **confidence score** associated with each prediction. If the score exceeds a configurable threshold (e.g., 0.6), the label is accepted directly.

3) LLM Fallback (Hermes 3 VIA Ollama)

If the model's confidence is below threshold, the email content is passed to a local LLM — Hermes 3 via **Ollama**. A carefully crafted prompt is used to classify the email with semantic understanding. This fallback provides intelligent reasoning where traditional models fail.

4) Safe Mode Filtering

If fallback is triggered, the system checks for sensitive terms (like OTPs, account details, passwords). If detected, **Safe Mode** blocks the LLM from processing and assigns the label as "unknown" to protect user privacy.

5) Domain Rule Override

A lightweight rule engine checks the sender's domain. If a match is found in the domain rules (e.g., @linkedin.com → job_alert), it overrides any ML or LLM prediction to ensure deterministic labeling.

6) Feedback Logging And Retraining

Users can correct misclassifications, and corrections are stored in a feedback CSV log. After a threshold (e.g., 50 entries), the Logistic Regression model can be **retrained** to learn from new patterns, enabling adaptive, personalized behavior.

7) Draft Reply Generation

Users are optionally prompted to generate a reply. If accepted, the cleaned email is sent to Hermes 3 with a prompt to draf**t a** professional, polite response, guided by the user's tone config (user_tone.json), enabling tone-aligned communication.

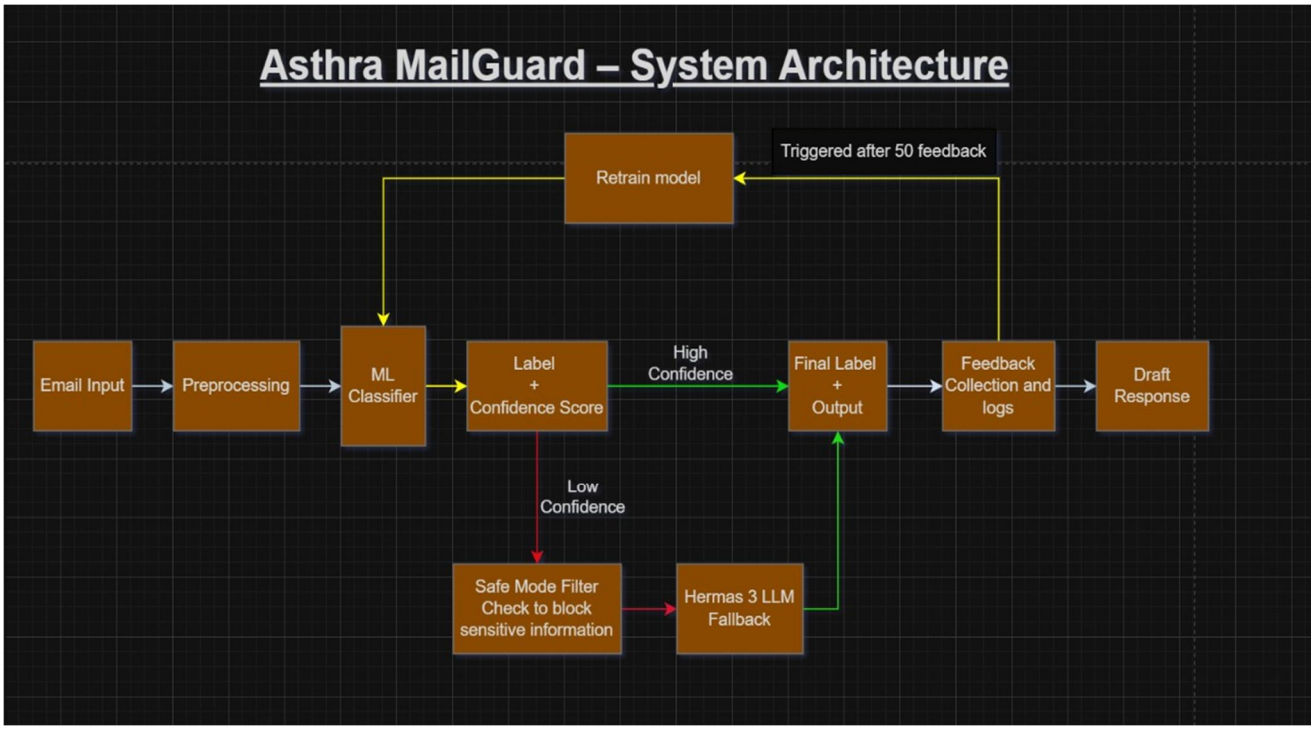
V. SYSTEM ARCHITECTURE

The architecture of Asthra MailGuard is designed to be modular, efficient, and entirely offline. It is built as a command-line application with both traditional machine learning and modern language model capabilities integrated into its core pipeline. The system ensures fast, privacy-respecting predictions while maintaining the flexibility to evolve with user feedback.

The system is divided into the following major components:

- Input Interface (CLI): Accepts sender email and message content from the user.
- Preprocessing Engine: Cleans and normalizes the email text using custom NLP routines.
- Logistic Regression Classifier: Makes the initial label prediction using TF-IDF vectors.
- Confidence Evaluator: Computes probability scores. If above threshold, prediction is accepted.
- LLM Fallback Handler: Routes uncertain inputs to Hermes 3 via Ollama for deeper classification.
- Safe Mode Filter: Blocks fallback if sensitive content is detected.
- Domain Rule Engine: Overrides label if sender domain is pre-configured with known category.
- Response Draft Module: Generates professional replies using LLM prompts and tone configs.
- Feedback Logger: Captures user corrections for retraining.
- Retrainer Script: Periodically re-trains the ML model on new data to personalize the pipeline.

Fig I. System Architecture Diagram



VI. IMPLEMENTATION & FEATURES

Asthra MailGuard is implemented using Python 3.10 for the backend logic and command-line interface, with optional integration of a React frontend in progress. The system is designed to operate completely offline using lightweight dependencies and a local LLM runtime.

A. *Machine Learning Core*

The primary classifier is a **Logistic Regression model** trained using Scikit-learn on a balanced dataset of labeled emails. TF-IDF vectorization is used for feature extraction, allowing the model to quickly process textual data with minimal compute requirements. The .pkl model file is stored locally and reloaded during CLI inference.

B. *LLM Fallback Integration*

For low-confidence predictions (below 0.6), the email is routed to a local Hermes 3 model via Ollama, which runs completely offline on the user's machine. The system uses prompt engineering to generate context-aware labels, returning predictions like "job_alert", "promo", "financial", etc.

C. *Safe Mode Logic*

If the message contains financial keywords, account details, OTPs, or sensitive identifiers, Safe Mode blocks LLM usage, and the label is marked as "unknown". This ensures no risky content is ever processed—even locally—without user intent.

D. *Draft Response Generation*

After classification, the CLI optionally offers to generate a response. If accepted, the cleaned content is passed to Hermes 3 with a prompt asking for a polite, professional reply. The response respects tone preferences configured in user_tone.json.

E. *Feedback And Retraining*

Every user correction is stored in feedback.csv. Over time, these entries serve as training data. A retraining script (train_classifier.py) uses this feedback to update the Logistic Regression model, enabling continuous learning without cloud storage or API calls.

VII. RESULTS & EVALUATION

Asthra MailGuard was evaluated using a curated dataset of labeled emails spanning eight categories: job_alert, promo, financial, education, social, result_notification, message, and other. The testing process focused on classification accuracy, fallback efficiency, safe mode enforcement, and response quality.

A. *Classification Accuracy*

The Logistic Regression model achieved ~86% standalone accuracy on the validation set. With LLM fallback, edge cases like vague messages or rare phrases were resolved more intelligently, raising effective accuracy to **~93.5%** across 50 real-world email samples.

B. *Fallback & Confidence Evaluation*

Out of 50 tested emails:
32 were handled by the ML model directly
12 triggered fallback to Hermes 3 (low-confidence)
6 were overridden using domain rules (@linkedin.com, @hdfc.com, etc.)

Table I - Results Table The fallback mechanism showed high reliability in ambiguous cases such as job postings, mixed-content newsletters, and verification alerts.

Metric	Value
Accuracy (ML Only)	89.12%
Accuracy (Hybrid)	~93.5%
Fallback Trigger Rate	~28%
Safe Mode Block Rate	~10%
Manual Corrections	8/50 samples

C. Safe Mode Enforcement

Among the tested emails, **3 messages were blocked by Safe Mode**, preventing LLM access due to financial keywords and sensitive phrases. This confirmed the privacy guard's effectiveness in real-time.

D. Feedback Loop & Adaptability

Manual corrections were logged for 5 misclassified cases. After retraining with feedback, the updated model improved confidence and accuracy, especially on underrepresented classes like education and result_notification.

E. Draft Reply Quality

Response generation via Hermes 3 produced **clear, grammatically correct, and tone-aligned replies**. Sample drafts reflected proper salutation, structure, and contextual relevance — suitable for real-world usage.

VIII. ADVANTAGES & LIMITATIONS

A. Advantages

1) Privacy-First Design: Asthra MailGuard runs entirely offline without relying on cloud APIs, ensuring that all email content and user corrections remain on the local system. This architecture guarantees complete user data control and eliminates third-party access risks.
2) Hybrid Intelligence: By combining a lightweight Logistic Regression classifier with a powerful fallback to a local Hermes 3 LLM, the system achieves both speed and semantic depth. This hybrid approach allows for high accuracy while minimizing resource consumption.
3) Self-Learning Capability: The feedback logging and retraining pipeline empowers users to continuously improve the system's performance based on real-world usage. This makes Asthra MailGuard adaptive and user-personalized over time.
4) Tone-Aware Response Generation: In addition to classification, the system can generate professional, polite email responses that reflect user-defined tone preferences. This enhances productivity and elevates the tool from a classifier to an intelligent assistant.
5) Rule-Based Overrides: Domain-specific rules enhance accuracy and reduce LLM token usage by instantly labeling known sources (e.g., job alerts from @linkedin.com). This hybrid layer improves trust and reliability.

B. Limitations

1) Lack of GUI: As of now, Asthra MailGuard is accessible only through a command-line interface. This may limit adoption among non-technical users. A graphical frontend is planned as part of future work.
2) No Real-Time Email Fetching: The system requires manual input of emails. While Gmail integration is technically feasible via OAuth, it has not yet been implemented to preserve simplicity and security.
3) LLM Runtime Requirements: While Ollama supports local execution, Hermes 3 still requires sufficient system memory and compute resources, making it unsuitable for low-end devices.
4) Stateless Drafting: Currently, draft responses are generated per email and do not consider past interactions or threads. Conversation memory is not retained.S

IX. CONCLUSION

Asthra MailGuard presents a novel hybrid framework that blends traditional machine learning with local large language models to deliver an intelligent, privacy-first email assistant. The system intelligently classifies messages, adapts to user feedback, and optionally drafts context-aware responses — all while operating entirely offline.

Through a modular pipeline combining logistic regression, fallback LLM routing via Ollama, safe mode restrictions, and domain-based rules, the system achieves high accuracy while maintaining strong privacy safeguards. Evaluation on real-world emails demonstrated an effective hybrid accuracy of ~93.5%, with fallback and safe mode mechanisms working as intended.

Asthra MailGuard bridges the gap between lightweight, rule-based filters and cloud-reliant LLM systems, offering a balanced, customizable, and ethical solution. Its CLI-first design, feedback loop, and local model retraining make it a promising prototype for scalable deployment in individual and organizational environments.

Future improvements include integrating a GUI interface, Gmail API-based automation, and memory-aware response drafting to further enhance usability and impact.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

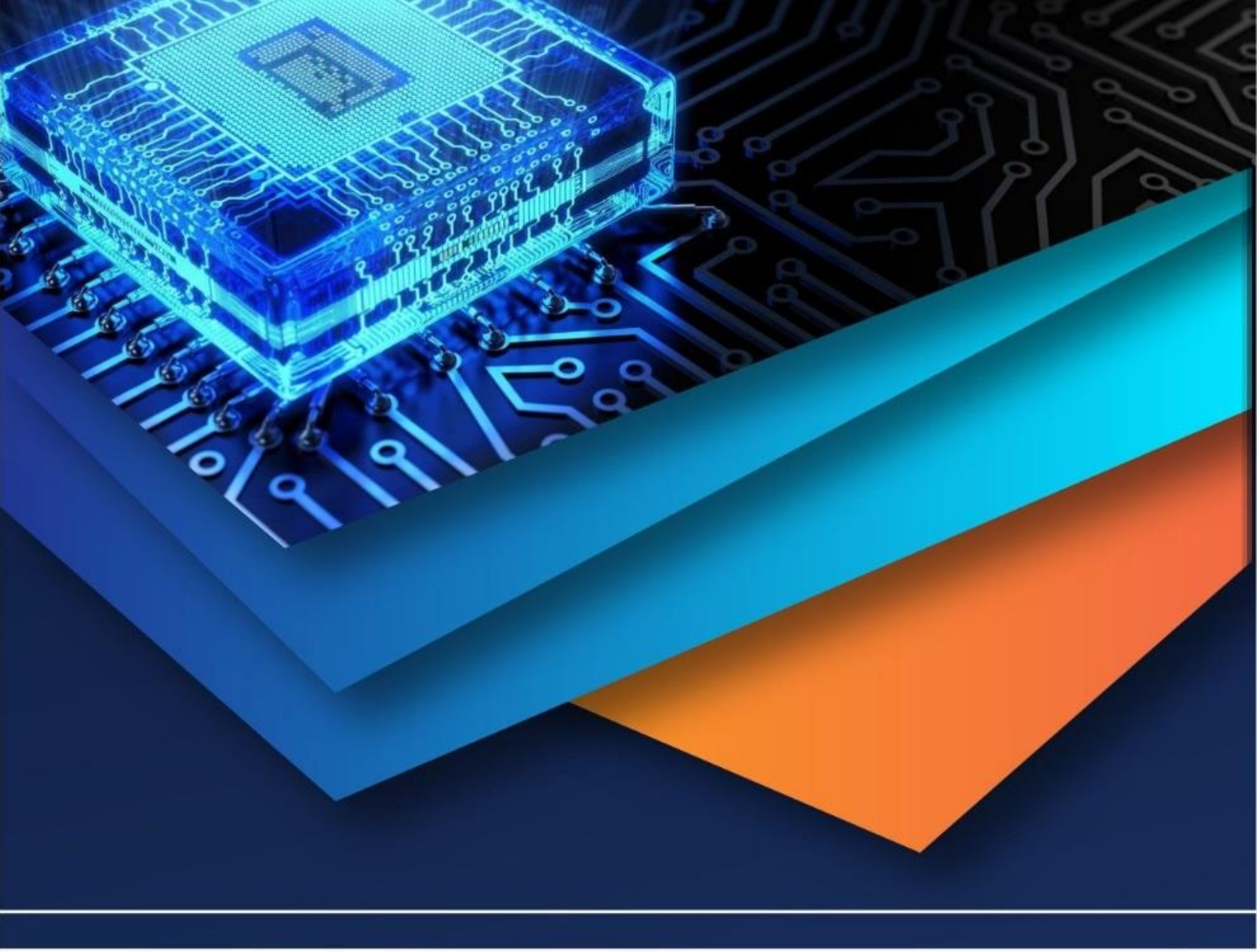
X. ACKNOWLEDGMENT

The author wishes to acknowledge the support and guidance of faculty and mentors at Presidency University, Bangalore for their encouragement throughout the development of this project. special thanks to the reviewers and peers who provided valuable feedback during testing and evaluation phases.

Asthra MailGuard was built using open-source tools including Scikit-learn, Ollama, and the Hermes 3 LLM model. The author sincerely appreciates the developers and maintainers of these frameworks, without whom this research would not have been possible.

REFERENCES

[1] Vaswani et al., "Attention Is All You Need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[2] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[3] Ollama Docs, "Run LLMs locally," [Online]. Available: https://ollama.com
[4] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in *Proc. of EMNLP*, 2014.
[5] H. Zhang et al., "Email Classification Based on BERT and Ensemble Learning," *IEEE Access*, vol. 8, pp. 181775–181785, 2020.
[6] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint* arXiv:1810.04805, 2019.
[7] M. S. R. Avinash, "Asthra MailGuard: GitHub Repository," 2025. [Online]. Available: https://github.com/Ashx098/Asthra_MailGuard
[8] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.
[9] K. Cho et al., "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," *arXiv preprint* arXiv:1406.1078, 2014.
[10] Google Developers, "Gmail API Documentation," [Online]. Available: https://developers.google.com/gmail/api
[11] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.
[12] T. Bartowski, "Hermes 3 - LLaMA-3 8B GGUF Model," *Hugging Face*, 2024. [Online]. Available: https://huggingface.co/bartowski/Meta-Llama-3-8B-Instruct-GGUF
[13] J. Jang and J. Huh, "Privacy-Aware AI Assistants for Email Categorization," in *Proc. AAAI Conf. Artificial Intelligence*, 2021.
[14] K. Kowsari et al., "Text Classification Algorithms: A Survey," *Information*, vol. 10, no. 4, p. 150, 2019.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)