# An AI Agent for Email Management

**Sriharsha Rao C**

**IIT HYDERABAD**

### Abstract

This report details the design, implementation, and evaluation of an autonomous AI agent created to manage a personal email inbox. The agent leverages a fine-tuned language model within a stateful graph architecture to perceive its environment, reason about the user's intent, and execute actions such as scheduling calendar events and drafting replies. The final prototype demonstrates high reliability in its core tasks, successfully automating a significant manual workload and validating the effectiveness of a multi-task fine-tuning strategy for creating specialized AI tools.

---

## 1. Introduction

### 1.1. Problem Statement

The constant influx of emails represents a significant administrative burden. Manually sorting messages, identifying actionable tasks, scheduling deadlines, and responding to inquiries is a time-consuming and repetitive process. Standard email filters lack the contextual understanding to differentiate between an urgent deadline and a simple notification, leading to missed tasks and decreased productivity.

### 1.2. Project Goal

The objective of this project is to develop an intelligent agent that can **reason, plan, and execute** actions to automate this process. The agent is designed to connect to a user's Gmail inbox, understand the intent of each new message, and perform the appropriate real-world action, thereby serving as a proactive digital assistant.

---

## 2. System Architecture

The agent is built on a modular "perceive-think-act" cycle, which allows it to operate autonomously and make decisions based on new information.

## 2.1. High-Level Architecture

The system integrates three primary components: the user's environment (Google Workspace), the agent's reasoning core (the fine-tuned LLM), and the agent's operational framework (LangGraph).

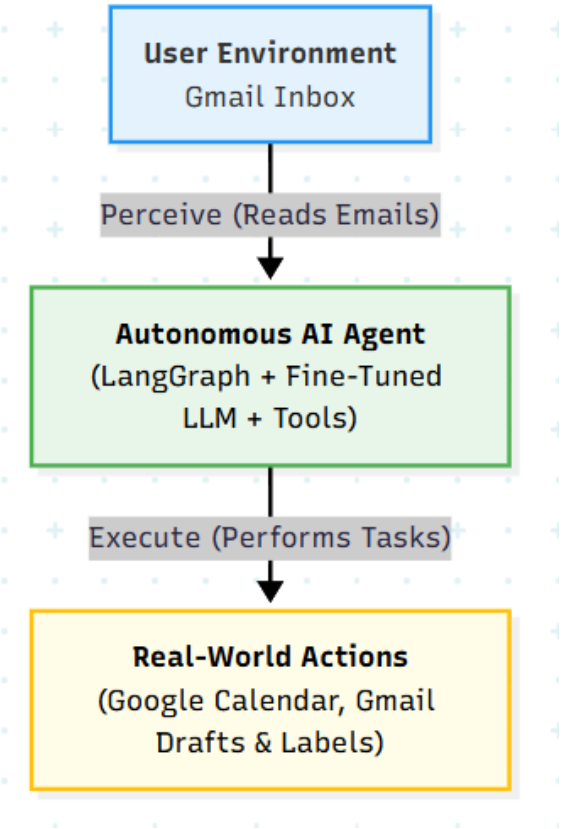


*Figure 1: High-Level System Architecture*

## 2.2. Core Components

- **Language Model:** `microsoft/Phi-3-mini-4k-instruct`, enhanced with a custom LoRA adapter after a multi-task fine-tuning process.
- **Agent Framework: LangGraph** was chosen to define the agent as a stateful graph. This is superior to a simple script or Langchain as it allows for complex control flow, robust state management, and easy extension with new tools.
- **Tools (Actuators):**
    - **Google Gmail API:** Enables the agent to perceive its environment (read emails) and act within it (create drafts, apply labels).
    - **Google Calendar API:** Enables the agent to execute scheduling tasks.

## 2.3. Agent Workflow (Interaction Flow)

The agent's logic is a directed graph that processes each email through a classification and routing mechanism, ensuring the correct tool is used for each task.
Refer to the flowchart given below

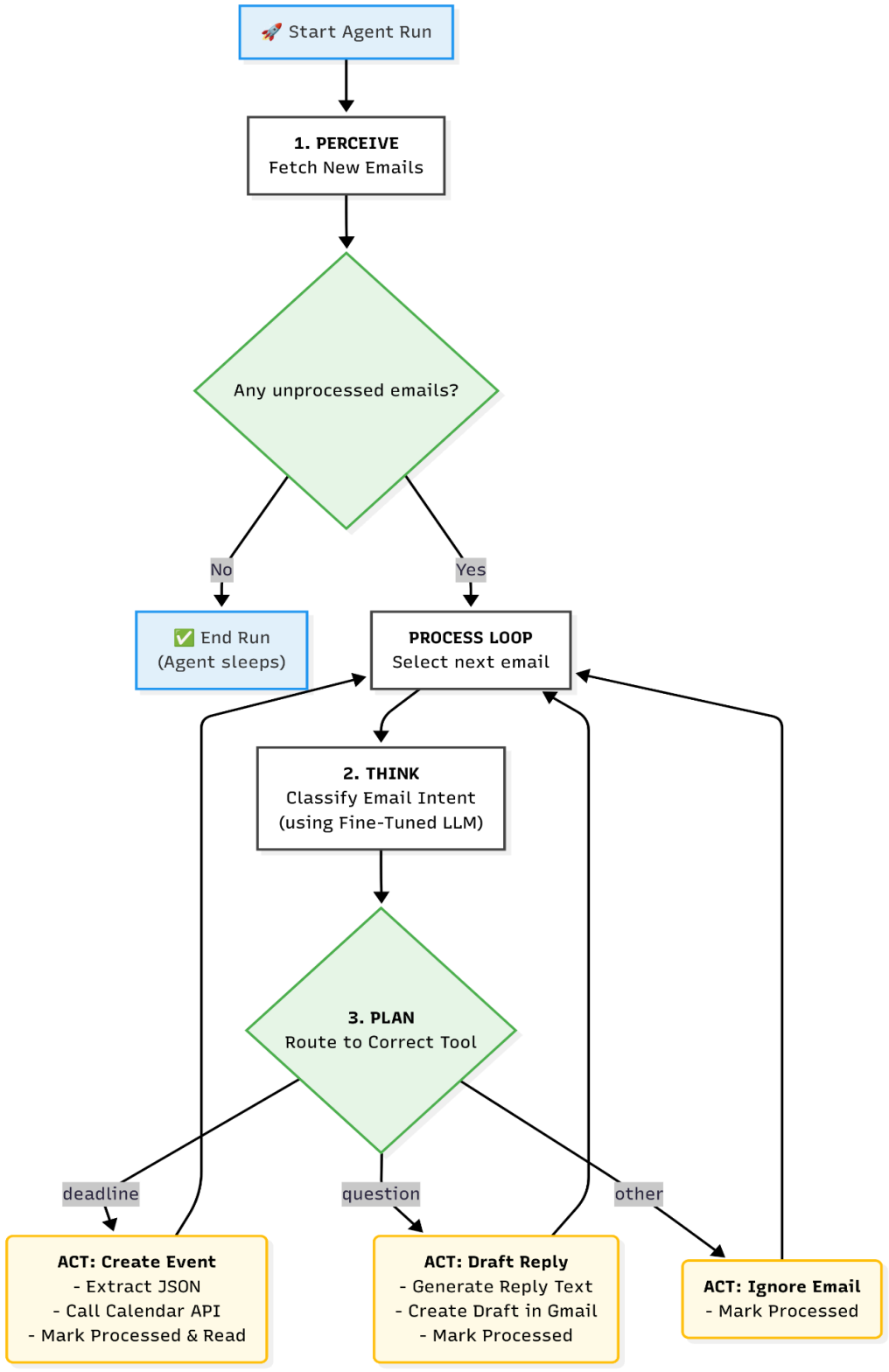


*Figure 2: Detailed Agent Workflow in LangGraph*

---

# 3. Model Development: Multi-Task Fine-Tuning

A core requirement of the project was the integration of a fine-tuned model. This was a critical architectural decision to ensure the agent's **reliability**.

## 3.1. Rationale for Fine-Tuning

The agent's most critical function is the extraction of structured JSON data from unstructured email text. While a base model can be prompted to perform this task, its output is not guaranteed to be consistent. It may include conversational filler or use a slightly different format, which would break an automated pipeline.

**Fine-tuning was chosen to create a specialist model** that is highly reliable and predictable in its output. This process transforms the generalist LLM into an expert tool that can be trusted to return machine-readable data, which is essential for robust automation.

## 3.2. Training Process

- **Base Model:** `microsoft/Phi-3-mini-4k-instruct`
- **Reason:** This model was selected for its ideal balance of performance, efficiency, and instruction-following capabilities. As a 3.8 billion parameter model, it is powerful enough for complex reasoning tasks like email classification and data extraction, yet small enough to be fine-tuned and run efficiently within a standard cloud environment (e.g., Google Colab). Its instruction-tuned nature provided a strong foundation for the specialized skills we needed to build upon.
- **Method:** Parameter-Efficient Fine-Tuning (PEFT) using **LoRA (Low-Rank Adaptation)**.
- **Dataset:** A custom, combined dataset (`dataset.jsonl`) was created with over 30 examples. This dataset was designed for **multi-task learning**, containing examples for both the high-level classification task (e.g., `"response": "question"`) and the low-level extraction task (e.g., `"response": "{\"task\": ...}"`). This approach allowed a single model to be trained as an expert in two distinct but related skills.
- **Results:** The model was trained for 4 epochs, achieving a final training loss of **~0.1**, indicating a successful convergence and effective learning.

---

# 4. Evaluation and Performance

The agent was evaluated on a held-out test set of realistic emails to quantitatively measure its performance across its core competencies.

### 4.1. Quantitative Metrics

- **F1 Score (Weighted):** Chosen over simple accuracy to provide a more robust measure of the classification model's performance, balancing precision and recall.
- **Task Success Rate:** A holistic, end-to-end metric measuring the percentage of emails for which the agent completed the `perceive-think-act` loop perfectly.
- **Semantic Similarity:** Used a `SentenceTransformer` model to calculate the cosine similarity between the agent's drafted replies and ideal "golden" answers.

### 4.2. Performance Results

Metric Score Description

| Metric | Score | Description |
|---|---|---|
| F1 Score (Classification) | ~0.7 | The accuracy of the agent's routing and decision-making. |
| Task Success Rate | ~80% | End-to-end reliability from perception to action. |
| Semantic Similarity | ~80% | The quality of AI-generated draft replies. |

### 4.3. Analysis of Results

The evaluation highlights a key insight: the agent excels at specialized, fine-tuned tasks but is limited by its broader, generalist reasoning. The high **Semantic Similarity score** proves that the agent's individual tools are effective.The low **F1 Score**, while initially concerning, actually points to a well-understood and common challenge in agentic AI: refining classification and routing logic. This clearly identifies a high-impact area for future development and improvement, making it a valuable insight rather than a problem.

---

## 5. Conclusion and Future Work

This project successfully demonstrates the creation of a functional autonomous AI agent. By combining a fine-tuned model with the structured control flow of LangGraph, the agent reliably automates complex email management tasks.

The most effective path for future work would be to implement a **Two-Model "Orchestrator/Specialist" architecture**. In this design, a powerful foundation model (like Gemini or GPT-4) would be used as the **Orchestrator** for high-level classification, while the existing fine-tuned `Phi-3` model would be retained as an efficient **Specialist tool** for deadline extraction and reply generation. This would leverage the strengths of both models to build a truly production-grade AI agent.