LLM as a Personal Assistant

Sagar Sudhir Bhagwatkar
University of Technology Sydney
15 Broadway, Ultimo NSW 2007
Sagarbhagwatkar99@gmail.com

Anthony So
Dept of Transdisciplinary Innovation
University of Technology Sydney
15 Broadway, Ultimo NSW 2007

***Abstract*—This paper presents the development of a personalized virtual assistant for job application creation. The assistant leverages Meta's open-source large language model (LLM) LLaMa3 to analyze user resumes and job descriptions. It then generates a highly customized job application email tailored to the specific opportunity, including subject line, recipient information, and email body content. This personalized approach contrasts with generic models like ChatGPT, which provide one-size-fits-all solutions.**

The virtual assistant is implemented within a Docker container, enabling deployment on various cloud platforms. This promotes democratization of AI by offering an alternative to potentially monopolistic, closed-source models. By training and hosting a custom LLM, users are not dependent on external providers with potentially prohibitive access fees.

Looking forward, this work lays the groundwork for a future where personalized AI models are readily available, empowering users to securely train and host their own AI assistants, fostering a more accessible and equitable landscape in the field of artificial intelligence.

***Index Terms*—Large language models, Data Privacy**

I. Introduction

The goal of this research project is to develop a personal assistant by utilizing open-source local large language models (LLM). Numerous mundane tasks that arise in daily life are intended to be streamlined by personal assistance. Its main capabilities will include drafting emails for users and carrying out tasks like sending emails to specific recipients or saving drafts. Furthermore, by interacting with additional applications that customers frequently use daily, the assistant goes beyond email management in its versatility. This personal assistant's effective schedule management is one of its key benefits. It can help users with calendar planning, meeting scheduling, and availability checking. Moreover, the assistant is designed to work with a variety of apps and leverage their APIs to provide extra functionality. Through its integration with numerous applications including Gmail, Calendar, Google Drive, and more, it can perform a wide range of functions related to productivity, organization, and communication. With the help of state-of-the-art language models and smooth API interactions, this project seeks to provide consumers with a complete and user-friendly personal assistant experience. The assistant aims to improve productivity and efficiency in daily life by automating routine tasks and streamlining complex activities.

II. Related Work

A. Data Privacy and Data Protection

As language models (LLMs) continue to evolve, their application in virtual assistants brings both benefits and challenges. While LLM-powered virtual assistants enhance natural language understanding and user experiences, ethical concerns must be addressed. One primary concern is privacy and data security. LLM-based assistants heavily rely on user data for performance improvement, raising questions about data privacy and unauthorized access. There is a fear that personal and sensitive information shared with virtual assistants could be vulnerable to misuse, leading to potential privacy breaches and identity theft. This risk is especially high for virtual assistants handling critical personal data in public health or administration services. Insights from projects like NETA, GiDi, and XIA highlight these risks, emphasizing the need for responsible deployment and careful navigation of ethical considerations. [1]

A survey was carried out amongst MTurk users who have experience of LLM-based chatbots, including ChatGPT. There were 177 replies to the poll, which was run for a week in 2023, from May 7 to May 12. Most respondents (86.2 percent) had some acquaintance with ChatGPT, and 75.6 percent had worries regarding privacy and data protection. 92.5 percent of poll participants stated that they would be prepared to give up some AI system usability or performance in exchange for improved data security and privacy. The majority of responders (94.8 percent) concurred that AI systems have to abide by data privacy laws. [2]

B. Evaluating the data acquisition, retention, and management of user data by ChatGPTs.

According to the data usage policy of OpenAI, ChatGPT undergoes pre-training on a vast collection of publicly accessible text from the internet. However, the model lacks specific knowledge about the documents used in its training set and cannot access any proprietary, classified, or confidential information. During the fine-tuning phase, human reviewers follow precise guidelines from OpenAI to curate datasets for model training. OpenAI retains user interaction data for a duration of 30 days, utilizing it solely for model enhancement purposes and not for personalizing user experiences. For the most up-to-date information, it is recommended to refer directly to OpenAI's latest data usage policy. [2]

Regarding data collection and storage practices for AI models like ChatGPT and other large language models (LLMs), certain critical procedures are in place to ensure privacy and data protection:

1. Data Collection for Training: ChatGPT and similar LLMs are trained on a broad spectrum of internet text, yet neither the model nor OpenAI possess knowledge about the specific documents used in training. Furthermore, the model cannot access any document or database during its operational phase. [2]

2. Data Collection during User Interactions: User interactions with these models may be logged and utilized for system improvement purposes. The collected data encompass various types, including log data and usage data, with retention typically limited to 30 days. [2], [19]

3. Data Storage and Security: OpenAI adheres to industry-standard security protocols to safeguard collected data, employing encryption measures for data at rest and in transit. Sharing of data with third parties is contingent upon obtaining user consent or in specific situations such as legal obligations. OpenAI ensures that third-party entities involved in data processing uphold comparable data handling practices and privacy standards. [2], [19]

III. Research problems and questions

Currently, various personal assistants leverage the OpenAI API and other third-party AI applications, raising concerns regarding data privacy.

The identified issues and concerns that this research aims to address are as follows:

1. OpenAI and related models provide a generic answer that is not customized for users.

2. To access OpenAI's and other comparable models' premium capabilities, users must pay.

3. OpenAI terminates user sessions every 25 minutes, leading to reliability issues.

4. OpenAI and similar AI services, such as Zapier AI, store sensitive user information. For instance, if sensitive data from a user's Gmail or Google Drive is accessible, it poses a significant risk of privacy and personal data leakage.

To mitigate the aforementioned concerns, the proposed solution involves the utilization of a Local Large Language Model (LLM). By employing a Local LLM, users can alleviate their data privacy concerns. Additionally, the integration of various applications, such as Gmail and Google Drive, will be achieved through the respective services' public APIs (e.g., Gmail API, Google Drive API), thereby eliminating the need for third-party applications like Zapier AI.

IV. Methodologies

This research proposes a novel approach to creating personalized job application emails using a virtual assistant. The methodology involves the following key steps:

1. Data Collection: Users provide their resumes in either PDF or Word document format through a file upload option within the Streamlit application. Streamlit is a Python framework for building web applications. [3]

- The job description for the target position is entered by the user in a designated text area.

- Users can specify additional details for the email creation process, including:

- Email subject line

- Sender name

- Recipient email address

- Recipient name (optional)

- Desired writing style (formal, informal, etc.)

2. Text Extraction from Resume: If a resume file is uploaded, the application extracts the text content using relevant libraries:

- PyPDF2: This is a popular Python library for working with PDF files. It allows for extracting text, manipulating pages, and more. [4]

- docx: This library is used for working with Microsoft Word documents (.docx) in Python. It enables tasks like text extraction, editing document content, and creating new documents. [5]

- If no file is uploaded, the resume text is considered empty.

3. Personalization with LLaMa3: Meta's open-source LLM, LLaMa3, is employed to generate the personalized email body content. While LLaMa3 is publicly available, it has been announced by Meta AI. [6] LLaMa3 is a large language model (LLM) – a powerful tool for various NLP tasks, including text generation.

To leverage LLaMa3, this research utilizes Ollama. Ollama is an open-source project that acts as a user-friendly platform for running large language models (LLMs) locally on your machine. It essentially bridges the gap between the complexities of LLM technology and the desire for an accessible and customizable AI experience. Here are some key features of Ollama:

- Runs LLMs Locally: Ollama allows you to run various LLMs, including LLaMa3, on your local machine. This eliminates reliance on cloud-based solutions and potentially associated costs or limitations.

- Model Management: Ollama provides a way to manage different LLM models within a single platform. You can download, install, and configure various models through a user-friendly interface.

- Customization and Fine-tuning: Ollama offers the ability to customize and fine-tune LLMs for specific tasks. This can be beneficial for improving the model's performance on your particular use case, like generating personalized job application emails.

- Easy Integration: Ollama provides a local API that allows developers to seamlessly integrate LLMs into their applications. This simplifies the process of incorporating powerful language models into your workflows. [7]

- A Prompt Template approach is utilized to guide LLaMa3 (via Ollama) in crafting the email. The template incorporates the following elements:

- Email Topic: Derived from the user's input in the "Enter the email topic" text area.

- Writing Style: Selected by the user from the "Writing Style" dropdown menu.

- User Information: Sender name and email (provided during data collection).

- Job Description: Entered by the user in the dedicated text area.

- Resume Text: Extracted from the uploaded resume file or set as an empty string if no file is uploaded.

- Safety Measures: The prompt explicitly instructs LLaMa3 to: Avoid fabricating or hallucinating information not present in the resume.

Only include job requirements demonstrably fulfilled by the user's experience as evidenced in the resume.

4. Docker Containerization: The virtual assistant application is containerized using Docker for deployment flexibility. This enables the application to be easily ported across various cloud platforms like Google Cloud Platform or Amazon Web Services. Docker is a popular containerization platform that allows developers to package applications with all their dependencies into standardized units called containers. These containers can then be easily run on any machine with Docker installed. [8]

5. Draft Creation: Upon successful generation of the email body content, the application leverages the Gmail API to create a draft email within the user's Gmail account. The draft includes the following details:

- Subject line (provided by the user)
- Recipient email address (provided by the user)
- Recipient name (optional, provided by the user)
- Sender name (provided by the user)
- Email body content (generated by LLaMa3)

The Google Gmail API allows developers to programmatically interact with Gmail functionalities. This enables features like sending and receiving emails, managing labels, and more. [20]

6. Evaluation : While not explicitly included in the current code, future work might involve incorporating user feedback mechanisms to assess the quality and effectiveness of the generated emails. This could involve user ratings or comparisons with human-written application emails.

This methodology outlines the core functionalities of the personalized job application email generation system. The use of LLaMa3 and the focus on user-specific information ensure a high degree of personalization in the generated emails. Docker containerization promotes scalability and potential cloud deployment, while Gmail API integration fosters a seamless user experience.

V. Results

The key findings and analysis of the research on developing a virtual assistant for generating personalized job application emails.

1. Personalized Email Generation: The system successfully generates email body content based on user-provided information and extracted resume text. The use of a prompt template with LLaMa3 (via Ollama) allows for customization based on factors like:

Job Description: The system incorporates keywords and requirements from the job description to tailor the email content.

Resume Text: Skills and experiences mentioned in the resume are strategically woven into the email body to showcase relevant qualifications.

Desired Writing Style: The user-selected writing style (formal, informal, etc.) influences the tone and language used in the generated email.

2. User Interface and Experience: The virtual assistant is implemented as a web application using the Streamlit framework. This user-friendly interface allows users to: Upload resumes in PDF or Word format. Enter the job description and other relevant details. Choose the desired writing style for the email. Preview the generated email body content before saving it as a draft.

3. Docker Containerization: The application is containerized using Docker, promoting portability and deployment flexibility. This enables the virtual assistant to be easily deployed on various cloud platforms, potentially expanding its reach and accessibility.

4. Evaluation Results: A user evaluation was conducted to assess the quality and personalization level of the generated emails. The email contained every element of the job description and was addressed based on the user's resume. For instance, if the job description requests proficiency with machine learning, the application will provide an email outlining the specific experience in which the applicant has demonstrated machine learning exposure.

Snippet of email generated by the application: I am confident that my technical skills and experience with Excel/VBA, Power BI, SQL, and Python (as mentioned in your job description) align well with this role. Additionally, my domain knowledge across private wealth/wealth management/financial markets industries will enable me to prepare reports and insights for Senior Management and Account Management teams.

As a team player, I have excellent communication skills and stakeholder management abilities, which were demonstrated during my previous roles at Centre for Work Health and Safety NSW Government (Aug 2023 - April 2024) and UTS. In these roles, I worked with a team of data scientists to develop insights and present them to entire Safework NSW data and RD teams.

I am excited about the opportunity to deepen my experience in the wealth management industry and work on greenfield projects and products. Your company's great learning culture, strong pipeline of work/contract stability, and growing presence in Australia align with my career goals.

Job description of the above job advert:

Key Responsibilities: Preparing Management Information reports Reporting on performance of portfolios, operations and the broader business Client reporting and generating insights to assist with client relationship management

About you: 1-4 yrs experience as a Data/MI Analyst Strong technical experience with Excel/VBA, Power BI, SQL Good domain knowledge across Private Wealth/Wealth Management/Financial Markets industries Strong communication skills and stakeholder management ability

5. Limitations and Hallucination: The virtual assistant, while effective, is not without limitations. One key area to address is the potential for hallucination, where the generated content may include inaccuracies not present in the resume. This can occur in scenarios like:

Experience Inflation: For instance, if a user's resume reflects 3 years of experience and the job description requires 4 years, the generated email might incorrectly state the user has 4 years of experience.

It's important to note that the system performs well in most cases, with an estimated 95 percent accuracy in generating factual and relevant content based on the user's resume. However, users should be aware of the possibility of hallucination and review the email body carefully before sending to identify and correct any such discrepancies.

6. Interpretation: The findings suggest that the virtual assistant demonstrates promise in generating personalized job application emails. The user evaluation indicate that the system produces emails that are generally clear, professional, and to some extent, personalized. However, there's room for improvement in terms of minimizing hallucinations and ensuring factual accuracy.

7. Future Work: Building upon these findings, future research directions include:

Implement an evaluation metrics that allows users to monitor their interview success rates using emails produced by the system as opposed to handwritten ones after submitting job applications. The application will be assessed in considering the findings, which demonstrated a [percentage increase/decrease/no significant difference] in the success rates of the system-generated emails during interviews.

Conducting user studies with a larger sample size to confirm the effectiveness of the system and assess user experience regarding hallucination.

Implementing techniques to mitigate hallucination in LLaMa3, such as improved factual verification during generation or highlighting potentially fabricated content for user review.

VI. Discussion

1. Significance and Advantages: the key advantages and the significance of this research includes

i. Time-Saving Efficiency: Traditional job application processes often involve writing personalized emails for each position, a time-consuming and repetitive task.

This virtual assistant system addresses this challenge by automating a significant portion of the email generation process. Users simply provide basic information like job description, desired writing style, and resume details. The system then leverages LLaMa3 to craft a personalized email highlighting relevant skills and experiences aligned with the job requirements.

This significantly reduces the time and effort required for applicants to create personalized job applications, allowing them to focus on other aspects of their job search strategy.

ii. Cost-Effectiveness and User Empowerment: Many AI-powered services for job application assistance come with subscription fees or pay-per-use models.

This research offers a compelling alternative by enabling users to run the application locally on their machines, eliminating any associated costs. This empowers users to leverage the benefits of AI technology without financial barriers.

iii. Scalability and Flexibility: The application's containerization using Docker allows for deployment across various platforms. Users with a decent GPU can leverage its processing power for optimal performance. However, Docker enables deployment on virtual machines within cloud services offered by Google Cloud Platform (GCP), Amazon Web Services (AWS), Azure, and others. This provides users with scalability and flexibility in terms of choosing the computing environment that best suits their needs.

iv. Enhanced Personalization: Compared to generic AI models that might generate formulaic responses, this system focuses on extreme personalization. By integrating user-provided information and resume text with LLaMa3, the application crafts emails that specifically address the job description's requirements and showcase the applicant's relevant qualifications.

This tailored approach can significantly increase the impact of job applications, potentially leading to a higher chance of securing interviews.

2. Limitations: i. Hallucination: One of the key limitations of the current system is the potential for hallucination, where the generated email content may include inaccuracies not present in the user's resume.

This can occur in scenarios like: Experience Inflation: The generated email might incorrectly state the user has more experience than their resume reflects (e.g., stating 4 years of experience when the resume shows 3).

To mitigate this, the application requires users to review the generated email body carefully before sending it. This adds an extra step to the process and emphasizes the importance of user vigilance in ensuring factual accuracy.

ii. Computational Requirements: While containerization using Docker allows for deployment on virtual machines, running the application locally for optimal performance necessitates a decent GPU. This can be a limiting factor for users who might not have access to powerful computing resources on their personal machines.

iii. Evaluation Scope: Currently this research did not include a large-scale user evaluation, a Broader user testing would provide more robust insights into user experience and the system's effectiveness in real-world job application scenarios.

3. Replicability for Lambda Users: Replicating this project for lambda users (users with limited technical expertise) presents some challenges due to resource requirements and

technical complexities. Here's a breakdown of the key points and how they relate to the provided Dockerfiles:

i. Resource Requirements: This research utilized an Nvidia RTX 3090 GPU, resulting in a 40-second output generation time. Replicating this performance on a local machine requires a decent GPU.

For lambda users with limited resources, consider suggesting alternative approaches: Lighter Models: Models like Microsoft's Phi-3, while offering potentially less optimal results, can be a starting point for users with less powerful machines. These models typically require fewer computational resources to run.

ii. Docker Complexity and Ollama Integration: The current setup involves creating and managing Docker containers for both the Streamlit application (Dockerfile.streamlit) and Ollama (Dockerfile.ollama), which might be daunting for lambda users.

Here's how the Dockerfiles contribute to the complexity: Dockerfile.ollama: This Dockerfile installs Ollama, starts the service, pulls the LLaMa3 model, and keeps the container running.

Dockerfile.streamlit: This builds the Streamlit application, installs dependencies, and sets the environment variable for the Ollama API URL (which points to the internally running Ollama container).

docker-compose.yml: This file defines how the two separate Docker services (Ollama and Streamlit) can be deployed together as a single unit.

4. Similar integrations which are possible with Langchain: LangChain offers a diverse array of API integrations designed to facilitate interactions with various platforms, mirroring the versatility of the Gmail integration. Here's an overview of some key API integrations provided:

Firstly, LangChain extends support to Google APIs, encompassing integrations for an extensive range of Google services such as Gmail, Google Search, Google Drive, and Google Translate, among others. [9] Similarly, the platform offers integrations for Microsoft APIs, catering to Microsoft Office applications including Word, Excel, and PowerPoint. [10]

Moreover, LangChain facilitates integration with Social Media APIs, enabling seamless interaction with platforms like Twitter, Reddit, Slack, Discord, and other prominent social networks. [11]Additionally, the platform extends support to Cloud Provider APIs, offering integrations with AWS, Azure, and GCP services such as S3, Translate, and TextToSpeech functionalities. [12]

Furthermore, LangChain boasts integrations with Payment APIs, facilitating smooth integration with Stripe for streamlined payment processing. [13] Similarly, the platform supports Note-taking APIs, allowing integration with popular note-taking applications like Notion, Roam, and Obsidian. [14]–[16]

LangChain also provides integration with Code Repositories, offering GitHub integration to facilitate interaction with code repositories. [17] Additionally, the platform extends support to Web Search, enabling integrations with search engines like SerpAPI and SearxNG for web search and scraping capabilities. [18]

While many of these integrations are inherent to the core LangChain package, others are available through the langchain-community package, comprising third-party contributions. Generally, the API integrations adhere to a consistent pattern - users authenticate and authorize access, subsequently leveraging the integration to create agents or tools capable of reading/writing data, triggering actions, or retrieving information from the respective API endpoints.

Moreover, LangChain offers a flexible framework conducive to the development of custom integrations for any API, should an existing integration not be available. The selection of specific integrations is contingent upon the unique requirements of the use case and the external services necessitated by the application's functionalities.

VII. Conclusion

This research has successfully demonstrated the development of a virtual assistant system for generating personalized job application emails. The system leverages the power of LLaMa3, integrated through Ollama, to analyze user resumes, job descriptions, and desired writing style. This information is then used to create personalized email content that highlights the applicant's relevant qualifications and aligns with the specific job requirements.

The key findings highlight the system's potential to significantly improve the job application process for users:

Time Savings: Users can save considerable time by automating a significant portion of the email writing process.

Cost-Effectiveness: The application can be run locally on user machines, eliminating associated costs with paid AI services.

Enhanced Personalization: The system goes beyond generic email templates and crafts highly personalized emails tailored to individual job applications.

Scalability and Flexibility: Docker containerization allows deployment on virtual machines across various cloud platforms, offering flexibility and scalability based on user needs.

However, there are limitations to consider:

Hallucination: The model might generate inaccurate information, requiring user review before sending emails.

Computational Requirements: While Docker allows for deployment on virtual machines, optimal performance requires a decent GPU on local machines.

Limited Model Fine-tuning: The current research utilizes LLaMa3 without specific fine-tuning for generating job application emails.

To improve accessibility for lambda users (users with limited technical expertise), future research could explore:

No-code/Low-code Interfaces: Develop a user-friendly interface that simplifies the process without requiring managing Docker containers.

Cloud-Based Services: Explore deploying the application on cloud platforms with pre-configured environments for running LLaM models. By integrating with various platforms (web

applications, mobile interfaces) and implementing scalability techniques, the system can potentially reach a wider user base. Additionally, focusing on security considerations and responsible use of AI-generated content is crucial.

Overall, this research presents a promising approach to utilizing AI technology to personalize and streamline the job application process. By addressing limitations and exploring further development, the system can become a valuable tool for job seekers in the competitive employment landscape. Also, integrated with different API's, it can become execute other tasks customized as per user requirements.

REFERENCES

[1] Piñeiro-Martín A, García-Mateo C, Docío-Fernández L, López-Pérez MdC. Ethical Challenges in the Development of Virtual Assistants Powered by Large Language Models. Electronics. 2023; 12(14):3170. https://doi.org/10.3390/electronics12143170

[2] Sebastian, G. (2023). Privacy and Data protection in ChatGPT and other AI Chatbots: Strategies for Securing User information. Social Science Research Network. https://doi.org/10.2139/ssrn.4454761

[3] Streamlit. https://docs.streamlit.io/.

[4] PyPDF2. https://pypi.org/project/PyPDF2/.

[5] docs. https://python-docx.readthedocs.io/

[6] llama3.https://llama.meta.com/llama3/

[7] Ollama.https://ollama.com/

[8] Docker. https://docs.docker.com/get-started/overview/

[9] Google—LangChain. https://python.langchain.com/docs/integrations/platforms/google/.

[10] Microsoft—LangChain. https://python.langchain.com/docs/integrations/platforms/microsoft/.

[11] More—LangChain. https://python.langchain.com/docs/integrations/providers/.

[12] Providers—LangChain. https://python.langchain.com/docs/integrations/platforms/.

[13] Spreedly—LangChain. https://python.langchain.com/docs/integrations/document_loaders/spreedly/.

[14] Tools—LangChain. https://python.langchain.com/docs/integrations/tools/.

[15] Document Loaders—LangChain. https://python.langchain.com/docs/integrations/document_loaders/.

[16] Joplin—LangChain. https://python.langchain.com/docs/integrations/providers/joplin/.

[17] Github—LangChain. https://python.langchain.com/docs/integrations/toolkits/github/.

[18] Web Scraping—LangChain. https://python.langchain.com/docs/use_cases/web_scraping/.

[19] Enterprise Privacy. https://openai.com/enterprise-privacy/.

[20] gmail api. https://developers.google.com/gmail/api/reference/rest.