AI PERSONAL ASSISTANT USING VISUAL-LANGUAGE RETRIEVAL-AUGMENTED GENERATION (RAG)

IBRAHIM BIN NASRUM 2116467

DEPARTMENT OF MECHATRONICS ENGINEERING
KULLIYAH OF ENGINEEERING
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
JUNE 2024

AI PERSONAL ASSISTANT USING VISUAL-LANGUANGE RETRIEVAL-AUGMENTED GENERATION

IBRAHIM BIN NASRUM

PROJECT SUPERVISOR:
PROF. DR. ABDUL HALIM

A REPORT SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR A DEGREE OF BACHELOR OF MECHATRONICS ENGINEERING WITH HONOURS

DECLARATION

I hereby declare that the project entitled "AI Personal Assistance Using Visual-Language Retrieval Augmented Generation" is the result of my investigations, except where otherwise stated, submitted to the International Islamic University Malaysia under supervision of Prof. Dr. Abdul Halim, Department of Mechatronics Engineering. I also declare that it has not been previously or concurrently submitted for any other degree at IIUM or other institutions.

………………………………

Ibrahim Bin Nasrum

Date: 12/6/2025

APPROVAL

I certify that I have supervised and read this study and that in my opinion, it conforms to acceptable standards of scholarly presentation, and it is fully adequate, in scope and quality, as Final Year Project report as partial fulfilment for a degree of Bachelor of Mechatronics Engineering with Honours.

……………………………………

Assoc. Prof. Abdul Halim Embong
Supervisor

ABSTRACT

In the era of rapid digital transformation, effective access to company knowledge has become a critical requirement for decision-makers. This project introduces a prototype AI personal assistant tailored specifically for C-level executives, such as Chief Executive Officers (CEOs), who require instant, reliable access to organizational data without relying on intermediaries. The assistant is powered by a Vision-Language Retrieval-Augmented Generation (VL-RAG) framework that combines image understanding and natural language generation capabilities, enabling it to interpret visual and textual inputs and generate accurate, context-rich responses. The core architecture integrates a vision-language model (BLIP-2) with a large language model (LLaMA 3 – 7B), augmented by a retrieval mechanism (FAISS) that indexes company data from diverse formats including Excel sheets, CSV files, and internal documents. The pipeline involves data preprocessing, semantic chunking, vector embedding, and similarity-based retrieval, followed by prompt-based response generation using the LLM. By leveraging multimodal data processing, the assistant can answer queries related to sales performance, inventory status, marketing strategies, and human resource allocation based on real company datasets. This system was evaluated using predefined query scenarios reflecting real-world executive needs. Quantitative metrics such as retrieval accuracy and response latency were analyzed alongside qualitative assessments of response relevance and coherence. Preliminary results indicate high accuracy and response quality, validating the system's capacity to serve as a reliable tool for executive decision-making. This research demonstrates the viability of a multimodal AI assistant in enhancing operational efficiency, reducing dependency on human intermediaries, and enabling informed strategic planning. The prototype also sets a foundation for future enterprise-level deployments where adaptive, vision-language-powered agents can streamline access to internal knowledge, transforming the way business leaders interact with corporate data.

ACKNOWLEDGEMENTS

In The name of Allah, the, the Most Gracious and the Most Merciful. By the grace of Allah SWT, I have successfully completed Final Year Project 1, "AI Personal Assistant Using Visual Language Retrieval Augmented Generation". Every opportunity, experience, and challenge overcome is a testament to His boundless blessings and a step towards creating positive change for those in need.

Besides, I would like to extend my heartfelt gratitude to Assoc. Prof. Abdul Halim Embong, my esteemed supervisor, for his invaluable guidance and mentorship. His expertise, insightful feedback, and unwavering belief in my potential have greatly contributed to the development of this project. I am sincerely grateful for the countless hours he invested in nurturing my academic growth and pushing me to excel.

My deepest appreciation goes to the dedicated lecturers who have significantly enriched my academic journey. Their commitment to teaching, sparking curiosity, and encouraging critical thinking has provided me with the tools and knowledge necessary to overcome challenges and succeed.

A special note of thanks to my family, whose unwavering support and sacrifices were my foundation throughout this journey. Their constant encouragement and joy have been my strength. To my friends, who have stood by me through every trial and triumph, your camaraderie, understanding, and occasional laughter have been invaluable. I am deeply thankful for your steadfast support and belief in my aspirations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

IIUM	International Islamic University Malaysia
AI	Artificial Intelligence
RAG	Retrieval-Augmented Generation
LLM	Large Language Model
BLIP	Bootstrapped Language-Image Pretraining
BLIP-2	Bootstrapped Language-Image Pretraining version 2
FAISS	Facebook AI Similarity Search
UI	User Interface
NLP	Natural Language Processing
NLU	Natural Language Understanding
NLG	Natural Language Generation
ML	Machine Learning
DL	Deep Learning
RNN	Recurrent Neural Network
Seq2Seq	Sequence-to-Sequence Model
MiniLM	Minimal Language Model (lightweight transformer model)
BERT	Bidirectional Encoder Representations from Transformers
LLaMA	Large Language Model Meta AI
QuilM-RAG	Quantized and Lightweight Multimodal Retrieval-Augmented Generation
CLIP	Contrastive Language-Image Pretraining
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
ETL	Extract, Transform, Load
ELT	Extract, Load, Transform
SQL	Structured Query Language
KG-RAG	Knowledge Graph-enhanced Retrieval-Augmented Generation

PEFT	Parameter-Efficient Fine-Tuning
RAM	Random Access Memory
OCR	Optical Character Recognition
CSV	Comma-Separated Values
PDF	Portable Document Format
JSON	JavaScript Object Notation
API	Application Programming Interface
DOC	Document
CEO	Chief Executive Officer
RQ	Research Question
VLMs	Vision-Language Models
BAS	Business Assistant Service
NER	Named Entity Recognition
KG	Knowledge Graph
Colab	Google Colaboratory (Cloud-based development platform)
Embedding	Vector representation of data (text/image)

CHAPTER 1
INTRODUCTION

1.1 BACKGROUND OF STUDY

In today's enterprise landscape, AI-assisted tools are no longer niche, they're essential. CEOs and top-level managers are increasingly embracing AI to streamline operations and improve data accessibility. A recent *Investopedia* report reveals that 79% of CEOs believe generative AI increases efficiency, and 55% are actively exploring or deploying AI tools in their organizations (Investopedia, 2024). This rising adoption underscores a growing demand for intelligent systems that can support leadership in accessing and interpreting company data without relying on intermediary staff

AI personal assistants are leading this digital transformation, with widespread global adoption across both personal and professional domains. According to *SEOSandwitch* (2024), over 4.2 billion people currently use AI personal assistants, and this number is expected to grow significantly as businesses seek smarter tools for workflow automation. The Hostinger AI Statistics Report (2024) reveals that around 56% of business leaders have reported early or moderate AI adoption, while only 20% have fully integrated AI across multiple business functions. This indicates that while AI experimentation is underway in many organizations, full-scale deployment across all operations is still in its early phases. For CEOs, who often manage large volumes of information and strategic decisions, these efficiencies are game-changing.

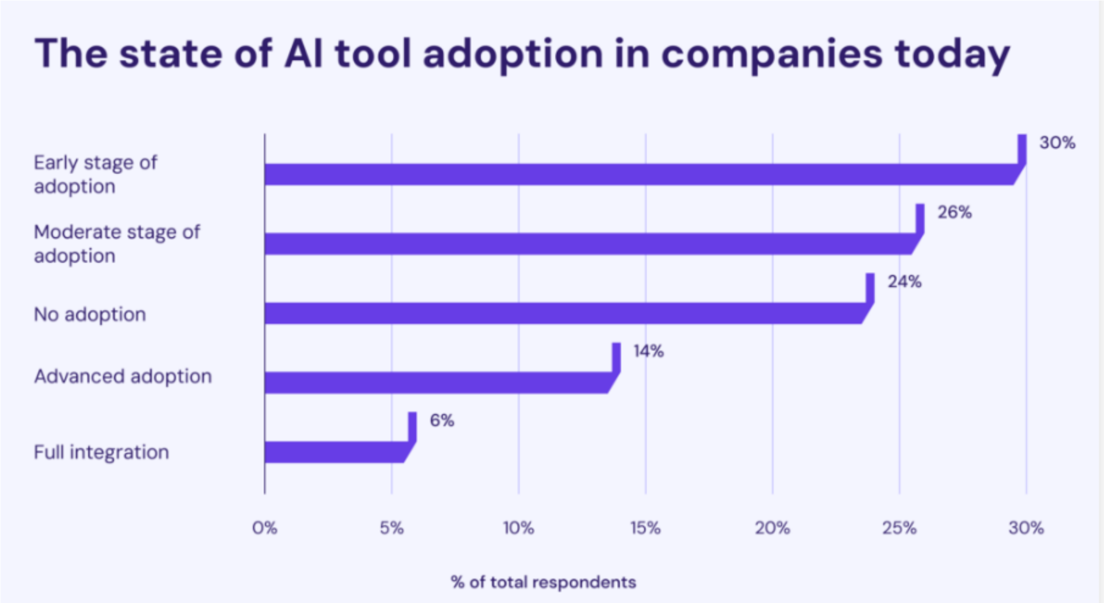

Figure 1.1: Percentage of around 56 percent business leaders report early or modearate AI adoption. Source: Salminen & Mauladhika, 2025 (Hostinger)

Looking ahead, the capabilities of AI agents are expected to evolve even further. *IBM* (2024) predicts that by 2025, AI agents will be fully autonomous in executing tasks, capable of interacting with complex business systems and making decisions with minimal or no human supervision. According to Hostinger (2024), despite concerns surrounding AI ethics and cybersecurity, the report also notes that 58% of companies plan to increase their AI investments in 2025, particularly in sectors such as logistics and customer service. These statistics illustrate a strong momentum toward AI implementation, reinforcing the growing demand for intelligent systems that streamline operations and support strategic decision-making.This trajectory highlights the urgent need for AI personal assistants that are not only conversational but also multimodal able to process both textual and visual data such as dashboards, reports, and scanned documents.

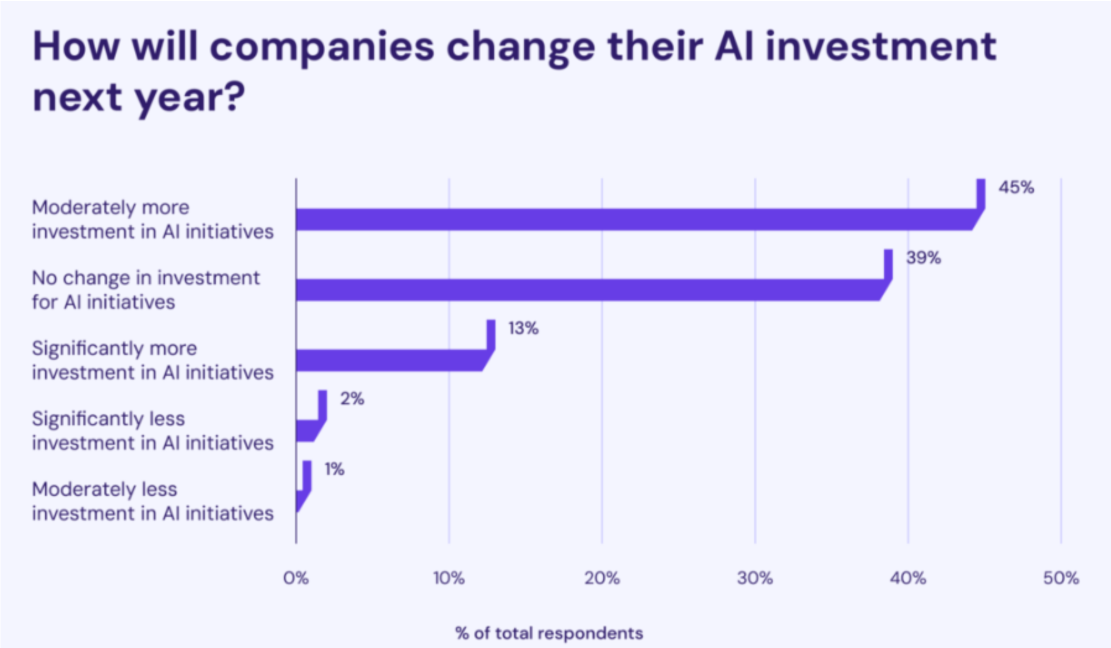

Figure 1.2: Nearly 60% of businesses plan to increase AI investments in 2025.
Source: (Salminen & Mauladhika, 2025 (Hostinger)

This situation underscores the need for a secure, multimodal AI assistant that understands text and images, retrieves internal data, and generates context-aware responses in real time. This project addresses that gap, offering an innovative solution tailored to the unique demands of executive information workflows.

1.2 PURPOSE OF THE STUDY

The purpose of this study is to design and evaluate a prototype of an AI personal assistant that integrates a vision-language model with a retrieval-augmented generation (RAG) framework. This assistant is intended to support company executives, particularly CEOs, in accessing internal data efficiently without relying on staff for manual retrieval. The short-term purpose is to demonstrate the system's ability to process both visual and textual inputs to provide accurate, context-aware responses. The long-term purpose is to encourage the development of more advanced, multimodal AI tools that can assist in corporate decision-making processes. It is hoped that this study will serve as a reference for future research and practical implementation of intelligent assistants in enterprise environments.

1.3 STATEMENT OF THE PROBLEM

Modern CEOs often face significant delays when accessing internal company information due to reliance on manual data retrieval by employees. Existing tools like dashboards or standard chatbots are limited in their ability to provide seamless access to both textual and visual data, reducing efficiency in fast-paced decision-making environments. Most AI assistants lack the multimodal capability to interpret charts, scanned documents, or structured tables, leading to incomplete or inaccurate responses. Furthermore, current solutions do not personalize output based on enterprise knowledge. This gap creates the need for an intelligent AI personal assistant that combines vision-language understanding with retrieval-augmented generation to deliver accurate, context-aware answers in real time empowering CEOs to make quicker, more informed decisions independently.

1.4 OBJECTIVES OF THE RESEARCH

1. To develop a functional prototype of an AI personal assistant that integrates a vision-language model for multimodal understanding.
2. To evaluate the system's decision-making performance based on a series of predefined user queries and real-world executive scenarios.
3. To optimize the system architecture for efficient information retrieval and low-latency response generation using open-source tools on limited computing resources.

1.5 SCOPE OF WORKS

i. Design and Implementation of System Architecture. This project includes designing a modular system architecture that integrates vision processing, language models, and retrieval mechanisms. The architecture ensures smooth data flow and scalability, suitable for deployment in real-world organizational environments such as executive dashboards.

ii. Development of Vision-Language Model Integration. The assistant incorporates a vision-language model like CoPalLi to process and interpret both image and textual inputs. This multimodal approach allows the system to extract insights from visual materials such as charts, dashboards, or uploaded company documents.

iii. Retrieval-Augmented Generation (RAG) Framework. The system employs an RAG framework using FAISS to retrieve relevant company data before response generation. This enhances factual consistency by grounding responses in internal knowledge bases, ensuring more accurate and contextually relevant outputs for CEO queries.

iv. Response Generation Using Llama 3. Llama 3 is utilized as the primary language model to generate responses based on retrieved context and user prompts. Its instruction-following capability and strong performance enable it to deliver executive-level answers with a high degree of coherence and relevance.

v. Evaluation and Testing with Real-World Scenarios. The prototype is tested using predefined CEO queries reflecting actual business needs. Metrics such as response accuracy, processing time, and user satisfaction are evaluated to measure the assistant's performance in decision-making support.

vi. User Interface and Interaction Layer Development. A web-based user interface is developed for seamless interaction with the assistant. It enables users to input queries, upload visual data, and receive real-time responses, making it accessible and user-friendly for non-technical executive users.

CHAPTER 2
LITERATURE REVIEW

2.1 INTRODUCTION

Since AI Assistant like ChatGPT or Deep seek has been viral in helping people find information, it has changed and redefined totally how individual, or an organization interact with information. Nowadays, AI assistants have been used recently in every sector, especially involving enterprise environment. Executives' top management like the CEO usually need to make a fast decision and acquire accurate information with context-rich real time data access to running their company effectively and efficiently. The literature review explores the current advancements in AI Personal Assistant specifically using Vison-Languange Models (VLMs) and Retrieval Augmented Generation (RAG) techniques. This review has been categorized into eight topics namely the AI Personal Assistants in Business and Enterprise, Natural Languange Understanding and Retrieval-Augmented Generation, Vision Language Models (VLMs) for Multimodal Understanding, System Architecture for AI CEO Assistant, Data Handling: Storage, Retrieval and Integration, Multimodal Processing and Response Generation, Ethical and Security Considerations and Research Gaps and Conclusion. By integrating this technology, it will help improve the quality of accessing data by the individual top executive like CEO. In summary, this chapter aims to provide a comprehensive overview of existing literature on AI-Personal Assistant, offering insights into the current state of research and development in this field.

2.2 AI PERSONAL ASSISTANT IN BUSINESS AND ENTERPRISE

Referring to Cambridge Dictionaries, enterprises mean an organization, especially a business or a difficult and important plan, especially one that will earn money. In the context of Enterprise AI Assistant, it means an AI that is being built personally to help the company. Business Assistant Service (BAS) is important in today's fast-paced economic environment as many modern entrepreneur's face challenges to analyze numerous information quickly with rapidly changing market conditions and pressure to make optimal decisions fast. (Ivashchenko et al., 2020).Because of the challenges, enterprises are increasingly in need of an AI assistant service to handle these challenges. Using AI technology, data can be grouped and processes from many sites to make conclusions and apply data driven decision making process (Haleem et al., 2022). According to (Marikyan et al., 2022), since the coronavirus (COVID-19) occurs, it has reshaped work pattern of organization to adopt remote working where digital assistant based on AI will be useful for work outcomes. Intelligence Personal Assistant has been built into mobile operating systems like Apple's Siri, Google Now and Microsoft Cortana to help user do tasks more efficiently every time (Goksel Canbek & Mutlu, 2016). (Ivashchenko et al., 2020) states that AI Assistant service main goal is to speed up, optimize and simplify entrepreneurs who have operated or just want to start the business in making decisions process. Furthermore, AI is also useful in marketing purposes as it can collect data and generate algorithms that will be useful in targeting the content for customers and employing the channel at the right moment (Haleem et al., 2022). From the point stated before, we can draw a conclusion and see that AI is useful in business to make decisions quickly and ease the process management with a short time. The use of AI Assistant in business is still in general which recently being used as an individual. The AI Assistant can be optimized better by specifically using company data to run the task more efficiently instead of using general open-source data to help make decisions.

2.3 NATURAL LANGUAGE PROCESSING (NLP)

2.3.1 NLP and Its Role in Enterprise AI Assistants

Natural Languange Processing is important as it serves as the backbones for understanding human or user language in AI assistant. Based on (Bharathi, n.d.-a; Cofino et al., 2024) Natural Language Processing (NLP) significantly essential technology that can be enhanced together with Artificial Intelligence (AI), Machine Learning (ML) and deep learning to improve human-computer interaction that will benefit in improving decision-making and operational effectiveness. NLP is also useful in transforming unstructured data such as chat, emails and social media post into more structured data. (Bharathi, n.d.). By applying transformation, enterprise AI assistants can understand and generate human-like response effectively in many business functions.

2.3.2 Broad Classification of NLP

NLP are classified into two categories which is Natural Language Understanding (NLU) which is human to machine and Natural Languange Generation (NLG) which is machine to human (Abro et al., 2023). According to (Su et al., 2020),the difference between the two is that NLG builds a corresponding natural language sentence based on the input semantics representation, whereas NLU extracts the semantics from the provided tests. Natural Language Understanding (NLU) in dialogue systems usually use semantic frames to understand what the user is saying. It can be done by analyzing user input to identify the domain, which is the main topic, predicting what user wants to do, and extract relevant information. In the past, the system handled each utterance in isolation which means look at each user message by itself, but now the system improves accuracy by incorporating context and recognizing speaker roles to enhance comprehension (Su et al., 2020a). Meanwhile, when the system knows what to say, Natural Language Generation (NLG) in dialogue systems will be used to produce clear and natural sentences in response based on the system's intended message. Older systems using Recurrent Neural Network (RNN) models lack flexibility and scalability, but in the latest system they use sequence to sequence (seq2seq) models which offer more diverse human response by learning directly from real conversation. (Su et al., 2020a).

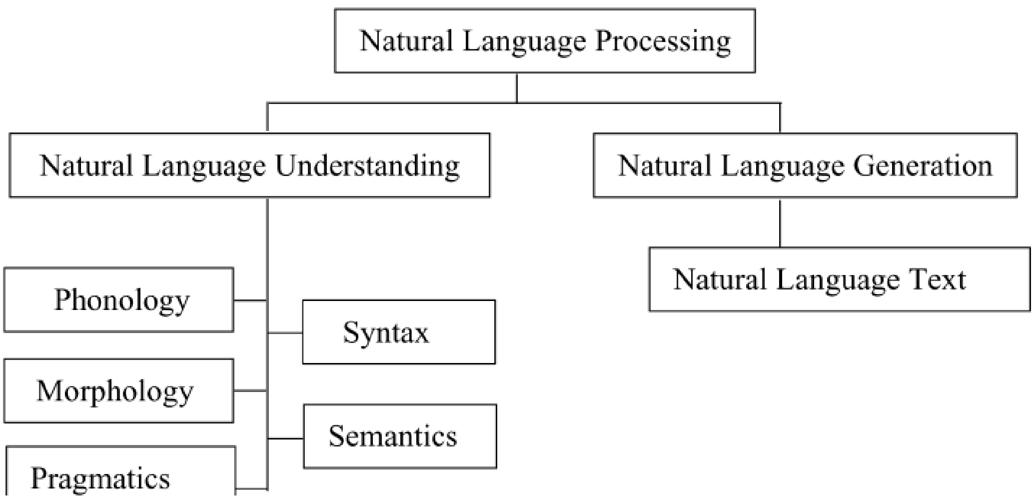

Figure 2.1: Broad Classification of NLP (Abro et al., 2023a; Solaimalai et al., 2024)

2.3.3 NLP Pipeline and how it works

The NLP pipeline is very important as it provides a structure framework that enables machines to process and interpret human language. NLP usually will start with text preprocessing. According to (Bharathi, n.d.-a; Supriyono et al., 2024a; Yagamurthy, 2023b), text preprocessing is the process on analyses the text to the level of individual sentences and words which includes cleaning and breaking down text using tokenization, segmentation, stemming and others. After that, the text representation process is very crucial to convert unprocessed text into a structured format that can be understood by machine learning more effectively (Supriyono et al., 2024a). Then, the feature selection process will identify important patterns by using filter-based techniques, followed by selection of models for training using machine learning algorithms(Su et al., 2020b). Finally, the trained model is deployed for tasks like sentiment analysis or translation. The model's performance will be evaluated and improved continuously. This systematic flow ensures accurate and efficient language understanding by computers.

2.3.4 Key NLP Techniques for Enterprise Applications

Human language is very complex and cannot be understood easily by machine. It is required to have a multiple level of processing to be understood by machine. This involves not only interpreting the meaning of individual words but also analyzing how those words and phrases relate to one another (Bharathi, n.d.). Because of this, Natural Language Processing (NLP) systems require to follow procedures to interpret natural text effectively. According to (Abro et al., 2023a), Deep Learning (DL), a subfield of Machine Learning (ML), is the foundation of most of these NLP systems. Figure 2 illustrates how DL is very useful and highly adaptable framework for describing the surroundings for both spoken and visual information. Common applications that are being used like machine translation, sentiment and emotion analysis, summarization, and NER may need to be understood properly to accommodate the system framework effectively. Based on (Bharathi, n.d.-b), machine translation plays a vital role in making conversion between language like tools of Google Translate. This is very crucial in breaking language barriers. Meanwhile, Named Entity Recognition (NER) is one key of information extraction in NLP which the main purpose is to detect and classify names, organization, locations, dates or numbers within a text.(Yagamurthy, 2023a). Besides, summarization is application involves synthesizing text that is usually found in search result into concise summaries to be used in academic databases and indexes. Furthermore, sentiment analysis is one of the most useful in NLP as it can identify emotion (Abro et al., 2023b; Pothuri & Varma Pothuri, 2024). For example, sentiment analysis is good in surveys, reviews and social media. In sum, the use in NLP has been completely enhanced text summarization method, but it still having some difficulty on comprehending the context which still require assistance with complex words and subtle nuances.

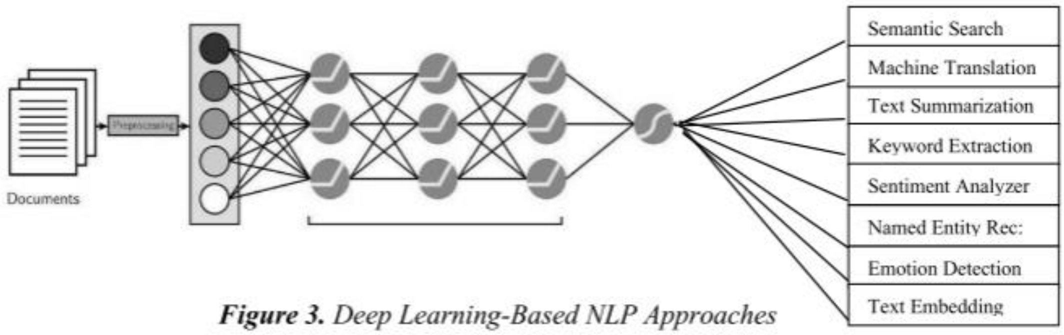

Figure 2.2: Deep Learning-Based NLP Approaches (Abro et al., 2023a)

2.3.5 Challenges and Current Trends

Natural Language Processing encounters some challenges, like dealing with ambiguity, context sensitivity, and the variety of human languages and dialects (Pothuri & Varma Pothuri, 2024; Supriyono et al., 2024b). As human language is too complex and sometimes have multiple meanings, it may need to interpret natural language truly, so the output becomes more accurate. The main challenge for NLU in NLP is that it is required for multiple roles like NLG rules as NLU having difficulty in interpreting natural language completely (Abro et al., 2023b). It’s also difficult doing text summarization as it deals with large amounts of meticulously labeled data and superior quality that take quite number of sources and time. (Supriyono et al., 2024b).

On the other hand, current trends have shown advancement and have improved understanding and generation tasks with the use of large language models (LLMs) like GPT and BERT. Bidirectional Encoder Representations from Transformers (BERT) is conceptual simple and empirically powerful, that are designed to pretrain deep bidirectional representation from unlabeled text (Devlin et al., n.d.). Besides, there's also growing interest in multilingual NLP, low-resource language processing, and explainable AI, aiming to make NLP systems more inclusive, transparent, and adaptable to real-world applications. Figure 3 shows the global trend in the application of NLP being used that increases rapidly from 2019 to 2025. We can see that text embedded is the most demanding application in NLP recently.

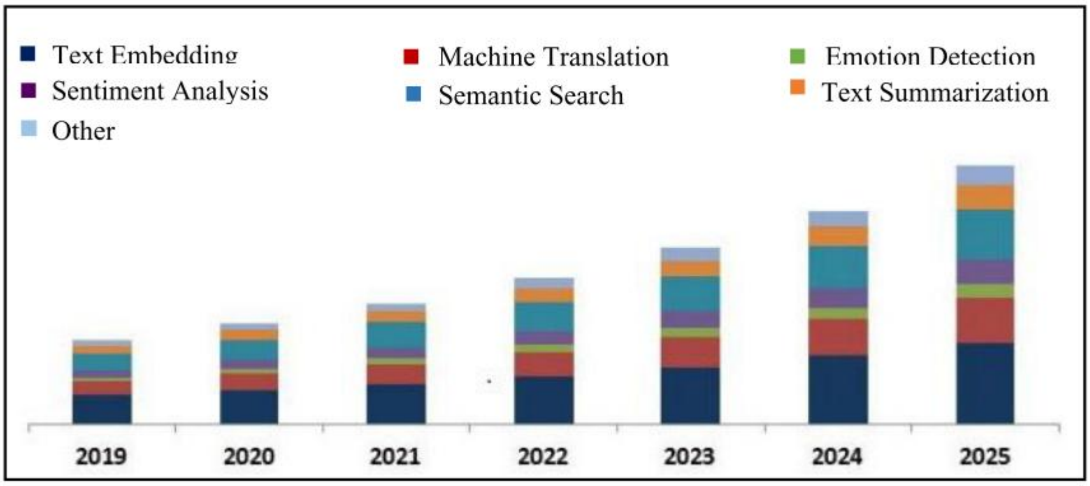

Figure 2.3: Analytics Of NLP Global Impact (Abro et al., 2023a; Solaimalai et al., 2024)

2.4 RETRIEVAL-AUGMENTED GENERATION

Retrieval Augmented Generation (RAG) is an approach used in Natural Language Processing (NLP) that merges the power of the pre-trained languange mode with support of knowledge obtain from the database data from open-source resources. (Bruckhaus, n.d.). RAG is also commonly divided into two important techniques, which information retrieval and text generation. For enterprises, RAG can help evaluate the data from the company and generate the best answers from the database. According (Packowski et al., 2024) , RAG is the best efficacious way in avoiding hallucinations and factual inaccuracy when answer questions while using the large language model (LLMs). In simple understanding, RAG is important to improve language models like ChatGPT or BERT-based models to answer questions or produce accurate and relevant text.

Three basic components in RAG are retriever, generator and knowledge base. The retriever acts as information retrieval and semantic search in searching or retrieving any relevant information based on similarity from the input query.(Bruckhaus, n.d.). After that, generators like GPT-4, Claude Opus or T5 which is a large pre-trained language model, will generate output accurately, consistently and contextually response. (Bruckhaus, n.d.) Furthermore, the knowledge base is a collection of structure and unstructured documents that can be collected from.(Bruckhaus, n.d.) In related with this research, it may be data of the company that we want to focus on. Below is the typical RAG architecture solution that has been discussed in the research of (Klesel & Wittmann,2025)

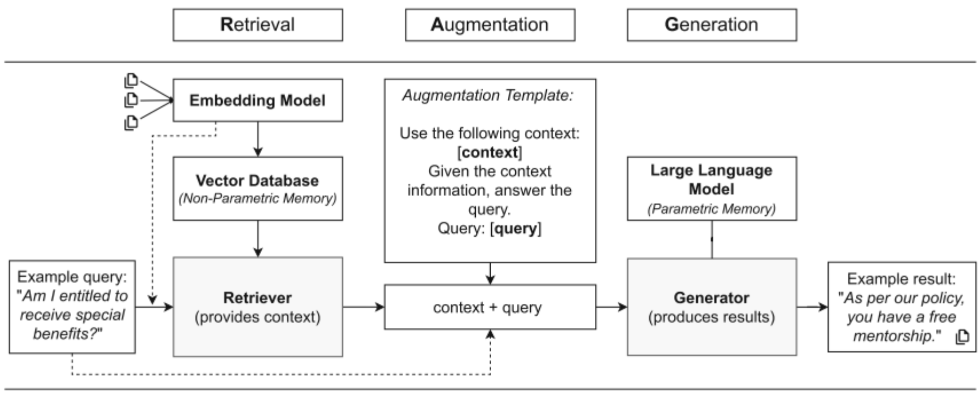

Figure 2.4: RAG Architecture Diagram (Klesel & Wittmann, 2025)

Meanwhile, there are findings that suggest the use of QuiIM-RAG (Advancing Retrieval-Augmented Generation with Inverted Question Matching) to reduce common problems like information dilution and hallucination that are usually faced by traditional RAG when dealing with large amount of structured data. (Saha et al., 2024) finds that the traditional dataset may not give a full alignment for user inquiry intention but by using the custom dataset and QuIM- RAG will help the LLM to be more accurate in avoid provide irrelevant information and unneeded details. Table 1 below shows the finding result between traditional RAG and QuIM-RAG that shows QuIM- RAG effectiveness.

Evaluation Matrix	Traditional RAG		QuIM-RAG	
	Traditional	Custom	Traditional	Custom
Faithfulness	0.69	0.72	0.91	1.00
Answer Relevancy	0.79	0.82	0.93	0.99
Context Precision	0.45	0.69	0.82	0.92
Context Recall	0.39	0.45	0.60	0.74
Harmfulness	0	0	0	0
BERTScore P.	0.32	0.37	0.55	0.63
BERTScore R.	0.29	0.35	0.63	0.71
F1 Score	0.31	0.36	0.59	0.0.67

Table 2.1: Comparison of performance metrics for traditional RAG and our novel QuIM-RAG using traditional and custom datasets (Saha et al., 2024)

However, some challenges that may be faced in implementing RAG in the enterprise sector are privacy, security and governance regulations that must ensure sensitivity of the customer never exposed or misused during the retrieval and generation process. (Bruckhaus, n.d.). These challenges may be overcome by strictly using the data of the company for the uses in the company only. Besides, others researcher finds that using RAG that involves large-scale applications may result in higher computational and financial costs. To overcome this research, I find that RAG can be implemented with open source LLMs and Haystack's orchestration that allow flexibility in development of scalable and financial cost applications. (Papageorgiou et al., 2025).

To sum up, Retrieval Augmented Generation is a potent method that combines retrieval and generation to improve the performance of language models. When working with real-world data, it makes models more accurate, current, and dependable. RAG is anticipated to be crucial to many real-world NLP applications as this discipline develops, ranging from customer service to healthcare and education.

2.5 VISION-LANGUAGE MODELS (VLMS) FOR MULTIMODAL UNDERSTANDING

Visual Language model is an additional system of the model to enhance its capabilities on having visual inputs. (Bordes et al., 2024). VLMs is designed to have visual inputs and textual inputs with ability to process and its ability to understand both types of data is called multimodal understanding. AI models have always been designed to deal with either text or images, but not at once. For instance, language models like BERT or GPT could comprehend and produce text, while computer vision models could identify objects in images. However, a lot of real-world activities, including describing an image, responding to a question about a chart, or reading text from a scanned document, call for knowledge of both input types. By learning from paired image-text data, vision-language models overcome this difficulty and can comprehend the connection between what is written or spoken and what is seen.

One of the popular VLMs is CLIP (Contrastive Language-Image Pretraining) by OpenAI which is an efficient and scalable method which is learning from natural languange supervision. (Radford et al., 2021) found that CLIP can outstand more than the best open public ImageNet in terms of computational efficiency, robustness and accuracy. In simple terms, CLIP can understand what is in an image and match it with the right text. For example, if given a photo of a cat, CLIP can choose the correct caption from many possible options.

Another powerful model is BLIP, which improves earlier models by using better image encoders and language decoders to generate more accurate captions and answers. (Li et al., n.d.) suggest using BLIP, which is a smart computer system that can learn from noisy image-text data. It contains MED as the brain and CapFilt which method to clean and improve the training dataset. (Li et al., n.d.) also find that, due to noise web text that recently occurred in visual languange learning, they have proposed CapFilt that has utilized web datasets in more effective ways.

GPT-4V (GPT-4 with Vision) and other more sophisticated VLMs go even farther. In a single discussion, they enable the model to process both text and visuals. For instance, a user can display a form and enquire, "Is this document filled out correctly?" or upload a graph and enquire, "What trend is shown here?" The model's ability to react by examining both the text and the visual structure demonstrates how potent multimodal AI has grown. (Qi et al., 2023) presents an in-depth comparative study of two pioneering models, which are Google Gemini and Open AI GPT-4V that both excel on interaction with humans, temporal understanding and assessments in both intelligence and emotional quotients. The findings show that both have excellent strength. While Gemini excels in providing comprehensive, in-depth answers with pertinent images and links, GPT-4V stands out for its accuracy and conciseness in responses. (Qi et al., 2023).

Optical Character Recognition (OCR) is another field associated with VLMs. OCR is an input device that is being used to read printed text.(Gomathy Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya et al., n.d.). VLMs can read and extract text from images, such as scanned documents, signs, or screenshots, using OCR tools like Tesseract or Google Vision API. To provide answers, summarize, or extract important information from visual data, this text can subsequently be integrated with language comprehension.

Furthermore, VLMs are strong because they are adaptable. They can be applied in many ways, such as assisting business users or CEOs in making choices through the analysis of papers, reports, and charts. VLMs give the AI personal assistant the ability to comprehend user-upload documents and photos in addition to responding to written queries. For instance, the assistant can process and intelligently respond when the CEO uploads a picture of a graph or a handwritten note and requests insights.

However, there are certain drawbacks to vision-language models as well. According to (Zhang et al., 2023), one difficulty is a data a fine-grained vision-language correlation modelling. There is still limited research exploring their effectiveness in dense prediction tasks like object detection and semantic segmentation. These tasks require fine-grained, local-level understanding between vision and language, which remains a challenge due to the scarcity of pre-training strategies focused on patch-level or pixel-level correspondence as a result, more research is needed to enhance zero-shot performance in such fine-grained visual recognition tasks (Zhang et al., 2023).

To sum up, vision-language models are essential for creating AI systems that have simultaneous "seeing" and "reading" capabilities. They enable AI personal assistants to do more than just read text, which makes them smarter and more practical in everyday situations, particularly in professional contexts where it's critical to comprehend documents, charts, and photos.

2.6 DATA HANDLING: INGESTION, STORAGE, RETRIEVAL, PROCESSING ND INTEGRATION

2.6.1 Data Pipeline

AI Personal Assistant needs good storage so that you can store the data generated when asked. Data must be handled correctly so the AI assistant can work fastly and accurately when to get access to both organized and unstructured data. The data handling will cover the data's storage location, retrieval method, and system integration. The data engineering pipeline, as Figure 1, shows the process involved in data management.

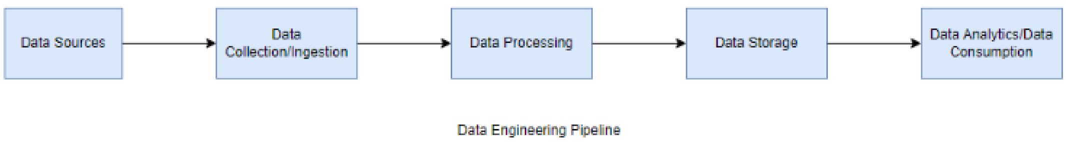

Figure 2.5: Data Engineering Pipeline (Balakrishnan, 2024)

2.6.2 Data Sources

Data is the fundamental thing needed for AI to work. Without enough data, AI cannot generate accurate and the best answer. There are studies from (Khan & Al-Badi, 2020) that classified data sources into four categories shows in figure 1. Social media network is the common real time unstructured data that can be analyzed further using NLP. Business and administration systems are commonly generated both structured and unstructured data from credit card activity or others. Besides, Government and administrative recently produced data like health, trade and population. Lastly, a ubiquitous system is collecting real time data from users through mobile devices and sensors. Additionally, data sources like HR databases, Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP) system also can be used to enhance ai assistant in decision making. CRM refers to tools or systems that companies use to manage interaction with customers while ERP is a system that helps manage company business processes in real time. (Jhurani, 2024) find that organization can achieve more advantages by integrating AI to enhance CRM functionality. So, from this we can understand that many sources can be used to be collected. For the company, the real data is needed in terms of both structured and unstructured data to ease and improve AI Personal Assistant.

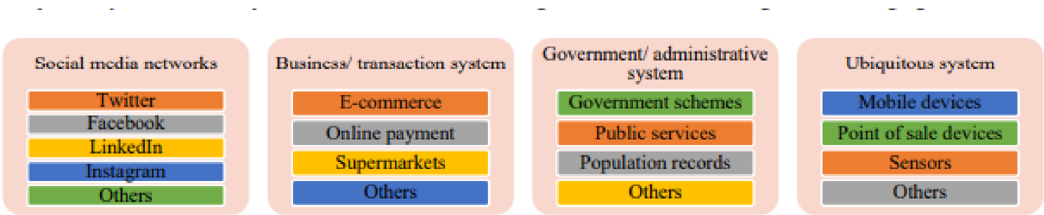

Fig 1. Classification of emerging data sources

Figure 2.6. Classification of emerging data sources 3. The Process of Decision Making Using Emerging Data Sources and Chall (Khan & Al-Badi, 2020)

2.6.3 Data Ingestion/Collection

Data ingestion is a process in data handling management which collects data from many sources to be processed, storage and analysis. (Alwidian et al., 2020; Elkina & Zaki, 2023.) . The goal of data ingestion is to make sure data is clean and stored appropriately and can be used easily. (Jhurani, 2024) suggest the use of RAG, RLF and fine-tuning that can give benefits to the performance. One studies state that data ingestion is an important process recently discussed with the concept of ETL which is extract, transform, and load (Alwidian et al., 2020). The data warehouse once implemented using ETL for managing structured data but when most of the data generated is unstructured data has changed the system by putting NoSQL in the system.(Elkina & Zaki, 2023.) . At the end, the new solution which is called data lake appears by using ELT (Extract, load and transform) which is more complex and requires longer execution time that offers more flexibility in handling many types of data. (Elkina & Zaki, 2023.) From this research, we can see that we can use ELT for data ingestion as for AI Personal Assistant is needed to ingest many types of data.

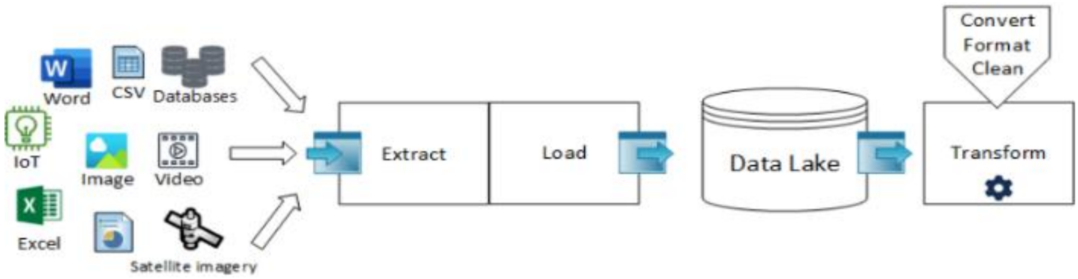

Figure 2.7: ELT process for Data Lake (Hamza Elkina & Taher Zaki, 2023).

2.6.4 Data Storage

AI Personal assistants in a company usually involve the use of the format like spreadsheets, databases, reports, infographic, emails and many more. Good data storage is needed to handle many types of data as shown in table below. For data storage traditionally use structured query language (SQL) for database management system. Typical SQL is PostgreSQL or MySQL. According to ("Balakrishnan," 2024), relational databases or cloud-based data warehouse like (Google Big Query, Amazon Redshift) is good in processing and analysis. But nowadays big data that keeps growing has evaluated the database management system that has spread the use of non-relational databases that store data in a non-tabular format (NoSQL) (Castro & Joy, 2024a). (Balakrishnan, 2024) also has stated that NoSQL databased like MongoDB, Cassandra) or object storage services like (Amazon S3, Google Cloud Storage) are good for unstructured data in terms of scalability and flexibility. NoSQL has become a good alternative because it can handle massive amounts of data with great scalability and flexibility. (Castro & Joy, 2024b). The experimental setup conduct by (Castro & Joy, 2024b) that used MySQL from SQL and MongoDB from NoSQL shows that MySQL performs efficiency for structure queries while MongoDB is greater more on handling large operation and unstructured data retrieval. Besides, case studies have given the best practical implication of database selection like for e-commerce companies that adopt hybrid architectures, it can use SQL databases for order processing and NoSQL database for product catalog management. (Castro & Joy, 2024b). So, in this research we require both types of structured storage such tables, row and column (SQL) and unstructured storage such PDF files, photos, scanned document to handle the data storage. We identified that we could use both SQL and NoSQL, but each has individual benefit and power that must be determined in which storage in this AI Personal Assistant to be used.

Data type	Examples
Structured	Last name, first name, age, address, registration code, … Exam score, number of registrants, …
Semi-structured	CSV, XML, JSON, …
Unstructured	Image, video, PDF, PPTX, DOCX, Streaming, …

Table 2.2: Type of Data (Hamza Elkina & Taher Zaki, 2023).

2.6.5 Data Retrieval

Using only data storage cannot make anything work, that's why data retrieval is important. From the data being collected, an AI assistant needs goods data retrieval to choose the most reliable data and information in a short time. In short, the data retrieval must be optimized for an AI assistant to generate an effective answer. Traditionally, data retrieval only uses keyword retrieval for text retrieval by using search engine to access information, but it can only work with structured data but struggle with semantic understanding. (Huang et al., 2020). Nowadays, we can use Retrieval-Augmented Generation (RAG) to look for any information or documents. RAG has enhanced the traditional retrieval information which uses keyword based. RAG is good because it is operated on pretrained parametric memory with non-parametric memory. (Klesel & Wittmann, 2025).

Besides, tools like (Facebook AI Similarity Search) FAISS and Chroma DB have been being used to support a more high-performance semantic retrieval. By using this vector databases will help in searching for similarity across millions of vectors embedding for millisecond. Researchers have found that there is a unique difference between FAISS and Chroma. FAISS is good as it has fast response time and gives optimal performance but as data size increases the advantages keep decreasing with the increasing of main memory. (Öztürk &Mesut,2024). The research also said that Chroma is more suitable to be used as support disk stored data that bring alternatives for larger dataset where memory constraint is more concern.(Öztürk & Mesut, 2024).

On the other hand, there is some research that has enhanced RAG methods. There are researchers that emphasized the benefit of using Knowledge Graph RAG (KG-RAG). A researcher has found that using KG-RAG has reduced irrelevant answer by over 50 percent and increase fully relevant answer by 88 percent compared to existing RAG. (Mukherjee et al., 2025). KG-RAG is enhanced by RAG that can offer more transparent reasoning path when required explanation. A full architecture of KG-RAG has shown in figure below.

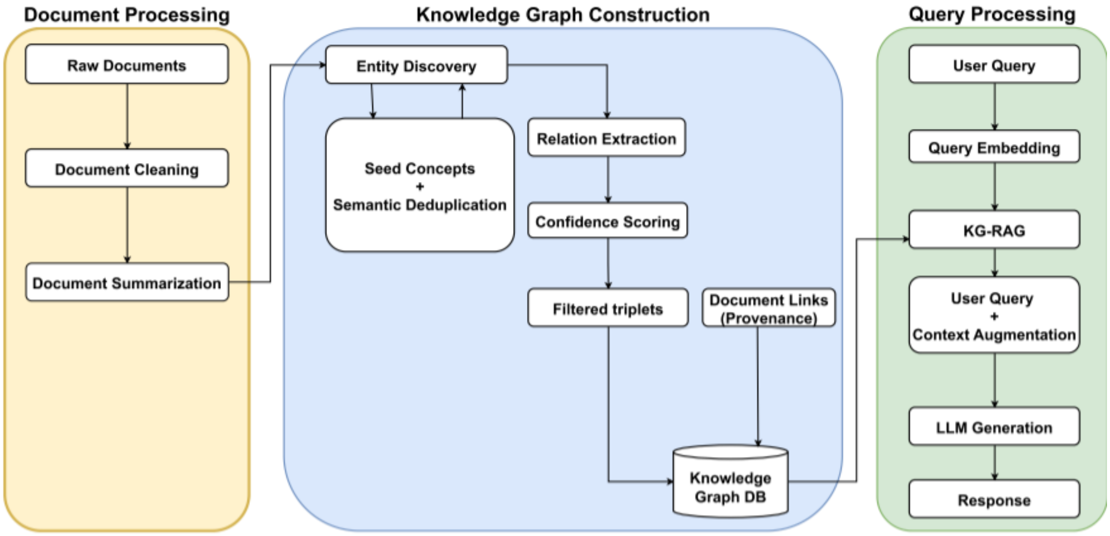

Figure 2.8: End-to-end pipeline for Knowledge Graph-based Retrieval-Augmented Generation (KG-RAG) (Mukherjee et al., 2025)

2.7 INTEGRATION OF OPEN-SOURCE LLMS AND TOOLS FOR AI APPLICATIONS

2.7.1 Introduction to Open-Source AI Ecosystem

Open-source AI Ecosystem is important in developing a more complex AI system that can reach AI singularity. (Okwu et al., 2024) mention that Meta AI and Open-source AI are outstanding because of its unique characteristics and its application. Open-source AI is open to the public in accessing the tools and artificial intelligence technology which are being created through collective effort from many developers (Okwu et al., 2024). (Vake et al., 2025) has found that open-source AI is beneficial as community of AI has rapidly improvised model and enhance model performance. So, Open-Source AI is very useful as it can help people to enhance and develop more AI Technology in the future.

2.7.2 Large Language Models (LLMs)

Large language model (LLMs) is game changer in making communication toward human language become possible by AI. LLMs are divided into two categories, which are open source LLMs which can be freely accessed and commercial LLMs with public APIs which is paid. Nowadays, there are many open source LLMs such as LlaMa(Meta), Mistral and Falcon that have been developed.

Llama (Large Language Model Meta AI) is an open-source model that has shown a strong performance and is optimized for efficiency and fine tuning. (Grattafiori et al., 2024)From Meta has found that Llama 3 delivers comparable quality to leading language model such as GPT-4 on a plethora of tasks. The research has observed that it performs competitively with the stated of the art on image, video and speech recognition tasks. Meanwhile, there are also a meta version of Llama 4 that states the future of multimodal AI that gives significant advancement in artificial intelligence, enhancing contecxtual understanding and performance. (Singh, n.d.).

Secondly, Mistral also was a good open-source model that was designed to outperform lareger counterparts while maintaining low latency which is suited for edge deployment and low resource application. One research from (Jiang et al., 2023), stated that the model was outperform Llama 2, Llama 1 in reasoning, mathematics and code generation.

Thirdly, Falcon, which was developed by the Technology Innovation Institute, was good for inference and training efficiency which was currently useful in multilingual and domain specific NLP tasks. Study from (Almazrouei et al., 2023) has stated that it has significantly outperformed models such as PaLM or Chinchilla and

improve the current model of LLaMa 2 or Inflection and even has perform above GPT 3.5 but below GPT 4 on the several tests conducted.
Besides this open-source free model, there are also several paid commercial/API Based LLMs like Open AI GPT-4, anthropic Claude, Cohere and Google Gemini.

Open AI was one of the most advanced LLMs available that was the main model behind ChatGPT. It supports complex reasoning, multilingual capabilities and multimodal input which make it outstanding for many applications. The study from(Roumeliotis & Tselikas, 2023) highlights that while ChatGPT is effective in generating Python code from prompts, crafting precise prompts and adapting the output still requires programming expertise, emphasizing that ChatGPT supports rather than replaces developers.

Besides, Atrophic Claude also was good LLMs model which designed with alignment and ethical safety in priorities. It excels in high stakes domain like law, customer service and others human centric values. Meanwhile of its benefit, the study from(Priyanshu et al., n.d.) has found that have some potential threats in privacy policies, the potential for hallucinations and biases in output and concern about third party data usage.

Lastly, Google Gemini was good in advanced search capabilities and knowledge grapht which are effective for knowledge retrieval and analytical tasks. According from (Imran & Almusharraf, 2024bhas stated that Gemini has potential revolutionary for future education technology as it has better problem-solving system.

In summary, these LLMs are all good and have individual strength. Open-source models allow users to custom and save cost while API-based solutions provide immediate scalability and performance.

2.7.3Frameworks and Toolkits

The widespread adoption of Large Language Models (LLMs) in real-world applications has been significantly supported by the emergence of purpose-built frameworks and development toolkits. These solutions simplify the integration of LLMs with various components such as external data sources, memory systems, and reasoning engines. They also assist in customization, deployment, and fine-tuning, enabling developers and researchers to streamline the process of building LLM-powered applications.

Firstly, LangChain is one of the leading frameworks tailored for constructing applications that require LLMs to interact with APIs, tools, and memory. It facilitates the creation of structured pipelines that include memory management, document retrieval, and prompt engineering. One research has explored and find that despite its strength, Langchain is emphasis on modularity and integration which affect complexity and potential security conscern. (Mavroudis, 2024)

Secondly, Haystack, an open-source NLP framework, is well-known for its use in semantic search and question answering tasks. It supports the combination of transformer-based LLMs with retrieval systems like Elasticsearch, FAISS, and OpenSearch. Haystack’s pipeline architecture allows for modular integration of preprocessing, retrieval, reading, and post-processing, making it ideal for building RAG-enabled enterprise solutions and domain-specific search systems.

Thirdly, AutoGPT and AgentGPT have popularized the concept of autonomous AI agents. These systems enable LLMs to autonomously break down objectives into smaller actions, plan steps, and interact with external APIs or tools. By iterating through goals and maintaining internal state or memory, these agents can complete complex tasks without continuous user prompts, showcasing early forms of goal-driven AI. One research from (Yang et al., 2023)has introduced an approach which the Additional Opinions Algorithm, which effective method that incorporates supervised based learners into the Auto-GPT scheme which enable lightweight supervised learning without need of fine-tuning.

Fourthly, OpenLLM, created by BentoML, focuses on the operationalization and deployment of LLMs in production environments. It provides an easy interface for exposing LLMs through REST APIs, while also offering capabilities like model versioning, observability, and performance monitoring. OpenLLM is compatible with major transformer models and supports both on-premises and cloud-native deployment scenarios. One researcher has examined the threat model and found that all the model leak query data, the user data that is queried at time. This research concludes that to achieve privacy preserving LLM that yields high performance and truly privacy preserving LLM adaptation must use open LLMs. (Hanke et al., 2024).

Besides, foundational libraries such as Hugging Face Transformers offer a wide collection of pre-trained transformer models suitable for various NLP applications. These libraries are crucial for both inference and training and are compatible with major hardware accelerators like GPUs and TPUs. SentenceTransformers builds on this by offering easy-to-use APIs for embedding tasks sucdh as similarity search, clustering, and ranking—especially useful for semantic retrieval and recommendation systems. Some researchers have said that hugging faces are good as give variety of applications and discussed in relation to its integration with others technology. (Pol, 2024)

Lastly, to support efficient model training and tuning, frameworks like Accelerate and PEFT (Parameter-Efficient Fine-Tuning) are often employed. Accelerate enables scalable training across multiple GPUs or TPUs with minimal configuration, while PEFT techniques such as LoRA help in fine-tuning large models without requiring full retraining, thus reducing compute costs and time. According to (Xu et al., 2023), PEFT offer effective solutions toward LLMs large size and computational demand by reducing the number of fine-tuning parameters and memory usage while achieving comparable performance of fine tuning.

Together, these frameworks and libraries enable the rapid prototyping, development, and deployment of sophisticated LLM-based applications. They empower innovation across diverse sectors including education, healthcare, legal analysis, and enterprise automation, transforming how intelligent systems are built and deployed in production settings.

2.8 RESEARCH GAP AND CONCLUSIONS

Numerous studies have examined multimodal AI systems, vision-language models (VLMs), and retrieval-augmented generation (RAG) in the literature. Nonetheless, most of these efforts concentrate on general-purpose applications like image captioning, customer support, and visual question answering. Research on the combination and customization of RAG and VLMs for corporate-level personal assistants, particularly for top management such as CEOs, is scarce. Additionally, there aren't many studies that address combining enterprise data privacy, real-time data retrieval, and customized multimodal processing into one solution. Most AI assistants currently in use primarily rely on either textual data or vision, rather than both in a task-specific and deeply integrated manner. This project is intended to close the gap in the creation of a safe, responsive, and context-aware multimodal assistant specifically for business decision-makers.

In conclusion, creating an AI personal assistant that combines retrieval-augmented generation and vision-language models has a lot of potential to increase job efficiency in a business environment. Important technologies like multimodal processing, RAG, and VLMs provide solid frameworks for creating an intelligent and responsive system. To manage real-time information retrieval and decision-making, robust system architecture must be created, but ethical and security issues must also be addressed. This project can make a significant contribution to the field of corporate AI by comprehending these elements and filling in the research gaps, developing a workable solution that helps CEOs obtain information in a timely and secure manner.

CHAPTER 3
METHODOLOGY.

3.1 SYSTEM ARCHITECTURE DEVELOPMENT WORKFLOW OVERVIEW

As this project aims to build an AI personal assistant that can help a CEO get information from company data in an easy way, the assistant must have capabilities in understanding both text and images input and give back a smart answer. This AI personal assistant must be designed in a multimodal AI system that can integrate with images and text. In this section will present the complete system overview for this AI assistant that will act as the foundation for system development in FYP 2.

The system architecture is composed into five important integrated modules which are user input handler, vision language encoder, document retrieval engine, large language model (LLM) generation engine and a unified chatbot interface. Each model is important as it plays critical role in transforming raw input into intelligence response that run in an open-source free platform only. The overall system diagram can be looked at at figure 3.1 below.

For the first step of the system, the user input module will receive either a textual query or an image upload from the user. This process will allow the user to enter their question or any following data to be asked in the chatbot. This module is responsible for validating the input format and directing the data into an appropriate pipeline. If the input is text, the system will pass directly to the retrieval engine. Meanwhile, if the input is an image example like business chart, the image will be processed first by the vision language module powered by BLIP 2 model to be performed the zero-shot image to text captioning and generate the natural language description of the image to be understood as textual query. By using this process, this will allow the system to interpret visual information without any manual OCR or tagging.

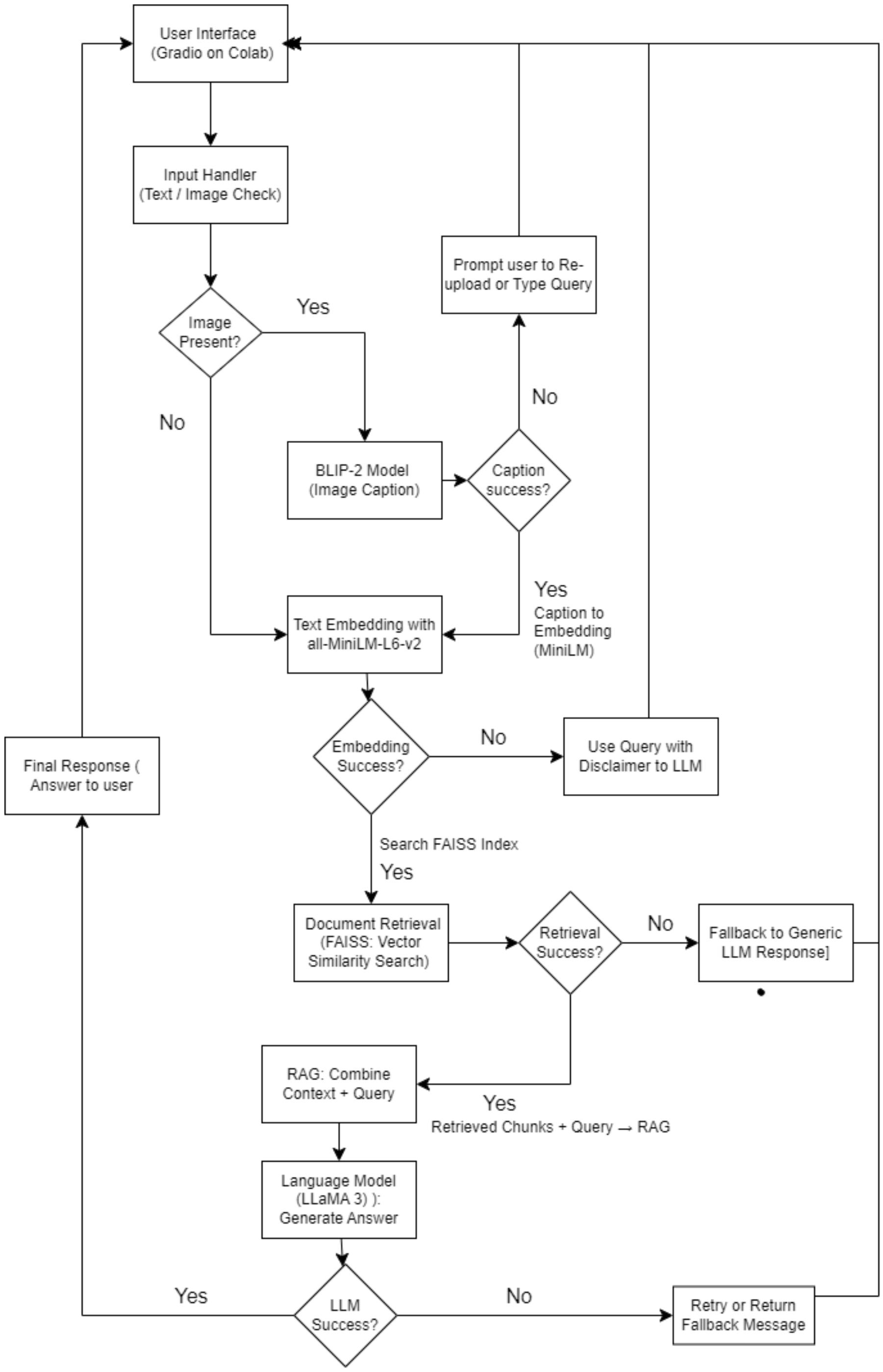

Figure 3.1: System Architecture AI Personal Assistant

Furthermore, after the text query has been established which either the text created by user or images that are converted to text, the text query will embed into a high dimensional vector space using all-MiniLM-L6-v2 which is a pretrained sentence transformer model created by Sentence transformer team. This function is to convert natural language sentences into numerical vectors (embeddings) that capture the semantic meaning of the sentence. Without this the system will not understand what the query means and cannot match the query with relevant information from company documents. This model will encode semantic meany of the query into vector q∈R384. By using a similar embedding method, it will be used to preprocess all company related documents (PDFs, reports, emails and etc.) which are chunked, embedded and stored in a FAISS. Using FAISS will allow the AI to quickly find the most relevant chunk based on cosine similarity between query vector and stored chunk vector.

After that, the retrieval model will perform vector similarity search to retrieve the top-k most relevant document chunks. The formula of cosine similarity score between query vector, q and document vector, d is shown below:

$$\text{cosine\_sim}(q, d_i) = \frac{q \cdot d_i}{\|q\| \|d_i\|}$$

Equation 3.1

The top 3 to 5 results with the highest similarity scores are concatenated to form a context-augmented prompt. This prompt is passed into the Language Generation Module (LLMs), which uses Llama 3 (7B) or Mistral 7B (depending on performance) hosted on Hugging Face and executed via Google Collab. The LLM uses this retrieved context to generate a coherent, factually grounded response.

Finally, the response is displayed to the user via a Gradio-based Chat Interface. Gradio provides a clean, browser-based UI where the CEO can type or upload, see answers, and interact conversationally. All backend processes (vision, retrieval, generation) are integrated and served through Google Collab notebooks, allowing easy testing, debugging, and iteration.

Module	**Tool/Model**	**Reason**
Image-to-Text	BLIP-2	Open-source V+L model with captioning
Embedding	all-MiniLM-L6-v2 (sentence-transformers)	Lightweight, fast on Colab

Vector Database	FAISS	Fast, scalable similarity search
Language Generation	LLaMA 3 7B (or Mistral 7B)	Accurate open LLM for factual answers
Chat Interface	Gradio	Easy to build and test in Colab
Development Environment	Google Colab /command prompt	Free GPU, easy deployment

Table 3.1: Tools and Frameworks for AI Personal Assistant

In summary, an AI CEO assistant's system architecture combines sophisticated AI models, clever retrieval techniques, and intuitive user interfaces. When everything goes according to plan, the assistant can serve as a trustworthy digital partner, assisting the CEO in keeping informed, saving time, and improving decision-making without relying on others for minor duties.

3.2 DATA SOURCES AND PREPROCESSING

In this project, the data source plays critical roles in influencing the AI assistant to gather company database and give effective responses toward the CEO's questions accurately. To simulate a realistic environment for easy application, this methodology will be going to be used in created structure set of synthetic data files that reflect a typical small business operation. For example, in this case we use a burger food stall. These files were designed to represent actual company documents like sales reports, inventory lists, staff schedules, marketing strategies and more. So, in this section we will discuss the source of data, and the preprocessing techniques applied. Besides, this section will discuss how the data is transformed into embeddings that are searchable by the assistant.

The data sources are usually given by the company's real dataset for its operation. As told above, the data in this project is generated across seven main categories to reflect various aspects of business operation. The files are organized as on table 3.2 below.

File Name	Content Description	Format
Sales.xlsx	Daily itemized sales records with revenue	.xlsx
Inventory.xlsx	Current stock and reorder threshold	.xlsx
IngredientsCost.xlsx	Monthly ingredient cost breakdown	.xlsx
StaffShifts.csv	Employee shifts and roles	.csv
MarketingCampaigns.csv	Campaign plans, budgets, and expected ROI	.csv
CustomerFeedback.txt	Complaints and compliments from customers	.txt
OperationsLog.txt	Notes on daily operations, cleanliness, repairs, etc.	.txt

Table 3.2: Data source type

These files on the table above simulate the information a CEO may want to ask about. For example, "What were the total sales on last Sunday?", "Which ingredient is running low?", or "How did the last marketing campaign perform?" After the question is asking the specific data is needed to obtain and analyze further to give the answer.

Before the data is retrieved to be generated, it must be going through the process of preprocessing. Preprocessing is a critical step that ensures all types of documents are brought into a clean and consistent format that can be used for semantic understanding and vector search. This project's three-step pipeline for data preprocessing aims to prepare and standardize a variety of input files for effective semantic search:

Step 1: Text Extraction

Text content extraction from a variety of file types is the initial step, Pandas.read_excel() is used to read Excel files (.xlsx), whereas pandas.read_csv() is used to read CSV files (.csv). Python's built-in file I/O operations are used to read the contents of plain text files (txt) line by line. This step's objective is to transform all data, regardless of file format, into a plain string format so that it may be used consistently as an input for the cleaning procedure that follows.

```
import pandas as pd

# Excel file
df_excel = pd.read_excel("data.xlsx")

# CSV file
df_csv = pd.read_csv("data.csv")

# TXT file
with open("data.txt", "r", encoding="utf-8") as file:
    text_txt = file.read()
```

Figure 3.2: Snippet code of converting into unified plain string

Step 2: Text Cleaning

Once the raw text is extracted, it undergoes a cleaning process to improve consistency and remove noise that might hinder semantic interpretation. The cleaning steps include converting all characters to lowercase to standardize the casing, removing punctuation marks such as periods, commas, and colons, and stripping away any special characters. Additionally, commonly used stop words like “is,” “the,” and “and” are eliminated to retain only meaningful content. Excessive or irregular spacing is also normalized. These cleaning operations result in a more uniform and semantically rich dataset, which significantly enhances the performance of the embedding model.

```
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

stop_words = set(stopwords.words('english'))

def clean_text(text):
    text = text.lower()
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    words = word_tokenize(text)
    words = [w for w in words if w not in stop_words]
    return ' '.join(words)
```

Figure 3.3: Snippet code of removing noise and preparing the data for vectorization.

Step 3: Chunking

After cleaning, the processed text is divided into manageable and semantically meaningful chunks. Each chunk contains between 100 to 200 words and is generated using a sliding window technique with a 20% overlap between consecutive chunks. This overlapping strategy helps preserve context between chunks and ensures that important semantic connections are not lost during segmentation. For instance, a text like "Today's burger sales were up 30% due to Instagram promo" may be chunked and represented as "burger sales up 30 percent Instagram promo," retaining the essential meaning in a compact form.

```
from sentence_transformers import SentenceTransformer

model = SentenceTransformer('all-MiniLM-L6-v2')
embeddings = model.encode(chunks)  # chunks from Step 3

def chunk_text(text, chunk_size=150, overlap=0.2):
    words = text.split()
    step = int(chunk_size * (1 - overlap))
    chunks = []
    for i in range(0, len(words), step):
        chunk = words[i:i + chunk_size]
        chunks.append(' '.join(chunk))
    return chunks
```

Figure 3.4: Snippet code enabling contextual understanding and splitting into overlapping chunks.

3.3 VISION MODULE IMPLEMENTATION

The vision module plays a key role in enabling multimodal understanding in the AI personal assistant system. While most traditional AI assistants rely solely on text input, the goal of this project is to allow the CEO to interact with the system using both text and visual documents such as photos of handwritten notes, whiteboard drawings, screenshots of Excel reports, or even scanned receipts. The vision module is responsible for extracting useful textual information from visual inputs and preparing it for semantic search and answer generation. This section outlines the implementation of the vision pipeline in terms of architecture, tools used, preprocessing steps, and integration with the Retrieval-Augmented Generation (RAG) pipeline.

The visual processing pipeline accepts image input (JPEG, PNG, or PDF), classifies it as either structured (like receipts, typed forms) or unstructured (like whiteboard photos, sketches), and chooses the appropriate extraction method. Structured data is processed via OCR, while unstructured data is passed to the Bootstrapped Language-Image Pretraining (BLIP-2) vision-language model.

We use a model vision which is BLIP-2 as it is a powerful open-source vision language model that can generate captions, summarize content or answer questions about images. This BLIP-2 is used only when the uploaded images are not suitable for simple OCR such as images containing a diagram, screenshot, or a handwritten message with illustrations.

$$BLIP(x) = \text{semantic text from image } x$$

Equation 3.2

The above shows simplified mathematical representation of how the BLIP model works in your system. Example to demonstrate how BLIP 2 works is like the input is a photo of a handwritten scribbled delivery note which is unstructured data. BLIP-2 is going to interpret the image and generate a natural-language description or semantic text based on its understanding of the image's content. The output instead of raw words, it returns a clear, structured sentence: *"List of items received: 5 buns, 2 packs of cheese, 3kg beef patties."*

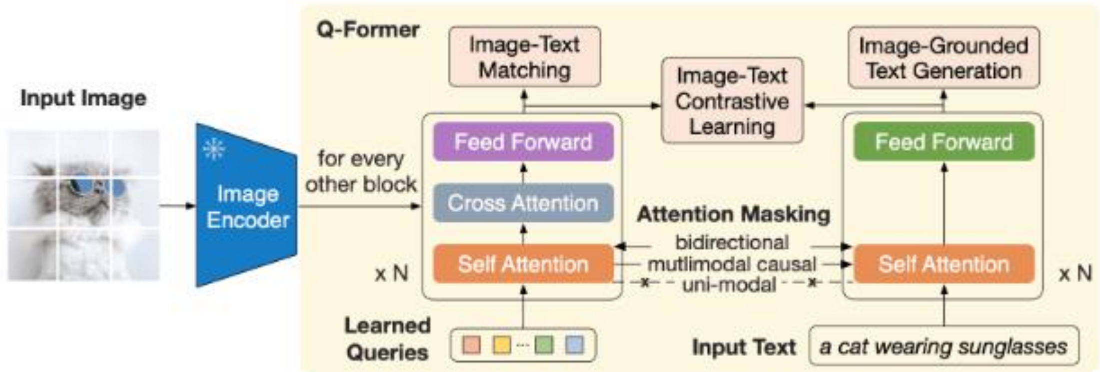

Figure 3.5: Architecture behind the BLIP 2 using Q-former transformer. Source: Li et al., 2023 (Hugging Face)

Meanwhile, when the system identifies a structured document (e.g., a receipt or typed table), Tesseract OCR is used. OCR is faster and more accurate on structured documents. The extracted text is processed as:

$$T(x) = \text{ExtractedText}, \quad \text{where } x \text{ is the input image}$$

Equation 3.3

The selected equation above is a simple mathematical expression representing the output of the OCR (Optical Character Recognition) process. Both OCR and BLIP-2 outputs are cleaned using regex and stop word removal. The cleaned output is then chunked (100–200 words, 20% overlap) and embedded using the all-MiniLM-L6-v2 model:

$$\vec{v}_i^{\text{img}} = f(E(C_i^{\text{img}})), \quad \text{where } C_i^{\text{img}} \text{ is a cleaned chunk}$$

Equation 3.4

This formula describes the process of converting a cleaned chunk of text extracted from an image into a vector embedding that can be stored and later used for semantic search. The table below shows how the model fallback logic happens in the system using both tesseract OCR and BLIP-2.

Condition	**Tool Used**	**Output Type**	**Fallback**
Structured text	Tesseract OCR	Plain text	If failed → BLIP-2
Unstructured / Diagram	BLIP-2 Vision-Language	Descriptive text	If failed → Ask user to re-upload
No readable text	—	—	Prompt user again

Table 3.3: Model Fallback logic

3.4 TEXT EMBEDDING AND RETREIVAL

In this project, text embedding and retrieval are essential components that allow the AI assistant to understand and find meaningful information across various company documents. These processes form the backbone of the Retrieval-Augmented Generation (RAG) pipeline, enabling the system to provide context-aware and fact-based responses to CEO queries. This section explains how documents are embedded, indexed, and searched effectively.

The main goal of embedding is to convert text into a numerical format that preserves its meaning. In this system, all cleaned document chunks are transformed into dense vectors using the free, open-source model all-MiniLM-L6-v2 from Hugging Face. Each text chunk, typically 100–200 words long, is first cleaned and normalized before being passed to the embedding model. This model converts each chunk into a 384-dimensional vector, known as embedding. These embeddings encapsulate the semantic essence of each chunk, allowing similar meanings to be identified even when phrased differently. For example, "the company's profits increased" and "the firm saw a rise in earnings" might result in similar embeddings due to their synonymous meaning.

Let:

$$C_i = \text{Cleaned text chunk}$$

$$E(C_i) = \vec{v}_i \in \mathbb{R}^{384}$$

Equation 3.5

The selected expression above simply defines the variable Ci as one unit of cleaned text that has been extracted from a document. Once in this cleaned form, CiC_iCi is ready to be passed to the embedding model (in your case, all-MiniLM-L6-v2) which converts it into a high-dimensional vector using the equation E(ci) above.

```
from sentence_transformers import SentenceTransformer

model = SentenceTransformer('all-MiniLM-L6-v2')
embeddings = model.encode(chunks)  # chunks from Step 3
```

Figure 3.6: Snippet code of using all-MiniLM-L6-v2 to convert each chunk into a 384-dimensional dense vector.

Once we have the embeddings, we store them using FAISS (Facebook AI Similarity Search), a free and powerful library for efficient vector retrieval. FAISS organizes the embeddings in a way that allows fast comparison using cosine similarity: Cosine similarity is used as the distance metric to compare vectors and retrieve the most semantically relevant chunks when a user query is received. Upon query submission—such as a question from a CEO—the query itself is also embedded using the same model. This embedding is then matched against all indexed vectors to find the topmost similar chunks. These relevant chunks are subsequently passed to a large language model (LLM) for generating coherent and contextually accurate responses.

$$\text{cosine}(\vec{v}_q, \vec{v}_i) = \frac{\vec{v}_q \cdot \vec{v}_i}{\|\vec{v}_q\| \|\vec{v}_i\|}$$

Equation 3.6

Where:

- vq is the vector of the query
- vi he vector of a document chunk
- The smaller the angle between vectors, the more similar the documents are.

```
import faiss
import numpy as np

dimension = 384  # embedding size for MiniLM
index = faiss.IndexFlatIP(dimension)  # inner product = cosine similarity if vectors are normalized

# Normalize vectors before indexing
normalized_embeddings = embeddings / np.linalg.norm(embeddings, axis=1, keepdims=True)
index.add(normalized_embeddings)
```

Figure 3.7: Snippet code of vector embeddings which are stored in a FAISS index to allow fast retrieval using cosine similarity.

3.5 RETRIEVAL AUGMENTED GENERATION PIPELINE

The Retrieval-Augmented Generation (RAG) pipeline is the core architecture of the AI assistant system developed in this project. RAG combines two powerful components retrieval and generation to answer user queries in a more factual, grounded, and intelligent manner. Instead of relying only on the large language model (LLM), RAG ensures that the answers are based on real data stored in company documents, making it ideal for assisting a CEO in decision-making. For this implementation, we utilize Meta's Llama 3 as the final generation model. The RAG pipeline is structured around three critical components: Context Retrieval, Augmented, and Generation

The first critical component is context retrieval. When a CEO submits a query For example, show me weekly burger sales affected by Instagram promotion the first thing the system needs to do is to encode the query and retrieve relevant information from the company documents. The user query is preprocessed using the equation below:

$$Q = \text{clean}(Q_{\text{raw}})$$

Equation 3.7

Where clean () is a function that performs a set of text preprocessing operations such as applies lowercase conversion, punctuation removal, stop word elimination, and whitespace normalization. After that, embedded be generated using the all-MiniLM-L6-v2 model:

$$\vec{q} = E(Q) \in \mathbb{R}^{384}$$

Equation 3.8

All company documents (PDF, Excel, CSV, TXT) have been chunked and embedded into a FAISS vector store. After that, a top- retrieval is performed using cosine similarity. The top k=5 results $Dk \cap Ds$ are returned for prompt construction. If fewer than 3 relevant documents pass a similar threshold, the system dynamically increases k or lowers T.

Parameter	**Value**	**Purpose**
k	5	Number of chunks retrieved
T	0.65	Minimum similarity score allowed

Table 3.4: Critical parameter used in the context retrieval

Furthermore, the second critical component is Augmented. Augmented means how to enhance generation by adding external knowledge like retrieved documents. Augmented work well as instead of just generating answers from its own pretrained knowledge, the model is augmented with up-to-date or company-specific info. So, the dataset has been used selective which focuses only on the dataset that we give that result the system to stay factual and can handle new data without retraining the model. In this context we use specific files like csv or any document from burger retail company for testing.

Thirdly, the crucial component in RAG is generation. The final prompt is sent to the Llama 3 7B model deployed via Hugging Face or local inference. The temperature is set low () to ensure determinism and factual consistency:

$$\text{Answer} = \text{LLaMA3}(\text{Prompt}; T = 0.2, \text{max\_tokens} = 256)$$

Equation 3.9

The LLM generates a coherent and grounded response based on the document context, not hallucinated knowledge. Example Output is according to the sales report from Week 21, Instagram promotion increased burger sales by 23%. Streaming or partial output can be implemented using asynchronous generators to improve latency.

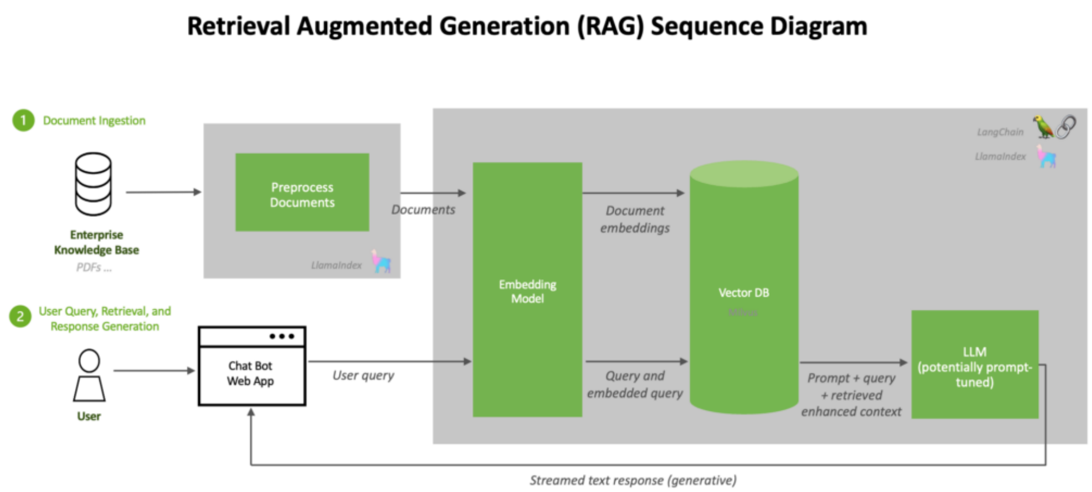

Figure 3.8: Overview of RAG pipeline components showing both ingestion and query flows. (Source: NVIDIA Developer Blog; Subramanya, 2024)

3.6 LANGUAGE GENERATION

In this project to implement the language generation component in the AI personal assistant system, the project employs Llama 3 (7B) from Meta AI, a state-of-the-art large language model that can be accessed through Hugging Face and executed on Google Collab with adequate GPU resources. The purpose of integrating Llama 3 is to generate coherent, contextually rich, and semantically meaningful textual responses when the assistant receives multimodal input (images + text) or solely text-based queries. This section outlines the rationale behind choosing Llama 3, describes how it is implemented, and evaluates it against other models such as Mistral, Zephyr, and GPT-J.

The Llama 3 series, especially the 7B parameter version, strikes a balance between model size, inference speed, and response quality, making it highly suitable for academic prototyping on platforms like Google Collab. It is trained on trillions of tokens and exhibits advanced understanding and generation capabilities, particularly when fine-tuned or used with prompt engineering and retrieval-augmented generation (RAG) frameworks. Compared to models such as Mistral 7B, which is known for its speed and lower latency, or Zephyr, optimized for instruction-following, Llama 3 demonstrates more robust long-context reasoning and semantic comprehension. Furthermore, unlike GPT-J, which is relatively older and less capable on contemporary benchmarks, Llama 3 excels in zero-shot and few-shot scenarios and is better at aligning with instruction prompts.

In mathematical terms, the generation G in a transformer-based language model like Llama 3 can be described as:

$$G(x_t|x_{<t}) = \text{softmax}(W_o h_t)$$

Equation 3.10

Where xi is the input token at time t, x smaller than t denotes all token precing xt, ht is the hidden state output of final transformer layer and Wo is the output projection matrix mapping hidden states to vocabulary logits. This probabilistic formulation allows the model to generate a sequence $Y = (y1, y2, \dots yn)$ where each token yi is sampled from the distribution $P(ui|y1, \dots, yi - 1)$.

The implementation in Google Collab is made efficient using Hugging Face's transformers library. Below is a simplified code snippet to demonstrate how the model is loaded and used for response generation:

```
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
import torch

model_id = "meta-llama/Meta-Llama-3-7B-Instruct"
tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id, torch_dtype=torch.float16, device_map="auto")

generator = pipeline("text-generation", model=model, tokenizer=tokenizer)

prompt = "Summarize the financial performance of Q1 2025 from the uploaded report."
response = generator(prompt, max_new_tokens=200, do_sample=True, temperature=0.7)[0]['generated_text']

print(response)
```

Figure 3.9: Snippet code on how the model is loaded and used for response generation.

In this setup, the assistant receives a high-level query, for example, a CEO asking about financial data trends, and can generate an insightful summary or answer that incorporates retrieved knowledge. The temperature parameter controls creativity, and max_new_tokens limits the response length for efficiency.

The model is also seamlessly integrated with the vision-language component such as ColPaLI or a similar architecture, where the output embeddings from image encoders are fused with textual prompts before being passed to the LLM. This cross-modal fusion empowers the assistant to interpret visual content, such as charts or diagrams, and respond in natural language.

In terms of architectural design, a retrieval-augmented generation pipeline is employed where relevant documents or image-text annotations are retrieved from an indexed database (using tools like FAISS) and concatenated with the input prompt to ground the model's generation in factual data. This approach mitigates hallucinations and enhances factual accuracy.

Model	Size	Speed	Strengths	Weaknesses	Best Use Case
LLaMA 3	7B, 70B	Medium (7B)	Strong reasoning, instruction-following, good long-context understanding	Requires high VRAM, slower on CPU/low-end GPU	Advanced chatbots, RAG systems, multimodal AI assistants
Mistral	7B	Fast	Lightweight, efficient, good for inference tasks	Slightly less accurate on complex prompts	Edge deployment, mobile inference, fast chat apps
Falcon	7B, 40B	Medium	Good general-purpose model, trained on refined datasets	Less instruction-tuned than LLaMA 3	Document summarization, search assistants
OpenChat	7B (Mistral-based)	Fast	Tuned for dialogue, aligned to user intent, good in Q&A scenarios	Smaller context window, chat-specific	Customer support bots, helpdesk assistants
Orca 2	7B, 13B	Medium	Emulates reasoning steps like GPT-4, distilled from teacher models	Requires better prompts, newer model so less community support	Research Q&A, educational tools, logical problem-solving bots
GPT-J	6B	Fast	Lightweight, easy to deploy, still useful for general text tasks	Outdated, weaker performance on current benchmarks	Simple automation, blog post generation, fast one-off inference
Zephyr	7B	Fast	Highly aligned, safe, chat-optimized, great prompt-following	Limited long-context ability, shorter context window	Safe chatbots, mental health bots, classroom tutors

Table 3.5: Different model LLMs capability

From the table above, the indicator of speed shows relative to size and resource usage where "Fast" implies it runs efficiently on a T4 or A100 GPU in Collab or similar environments. While strengths and weaknesses are benchmarked against tasks such as text generation, summarization, reasoning, and instruction-following. After that, the **best use case** aligns with practical deployment scenarios relevant to your AI personal assistant project.

Llama 3 (7B) is chosen for this project due to its superior instruction-following ability, strong reasoning performance, and compatibility with multimodal and retrieval-augmented frameworks. It delivers accurate, context-aware responses even in complex scenarios, making it ideal for a CEO assistant that must process both text and visual data. Its balance of performance and efficiency also allows for smooth deployment on platforms like Google Collab, making it the most practical and powerful open-source model for this use case.

Llama 3 can obtain by installing Ollama. The system requirement as below:

System Requirements

- OS: macOS, Windows, or Linux
- RAM: 8 GB minimum (16+ GB recommended for better models)
- CPU is fine, but GPU (NVIDIA) makes it faster (optional)

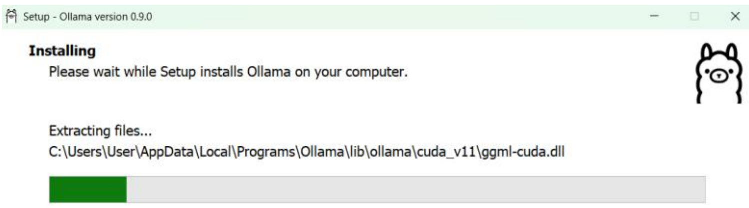

Figure 3.10: Installation process of Ollama for using Llama 3 model.

If the priority were ultra-fast response time on minimal hardware, it might lean toward Mistral. If safety in conversational alignment was the top concern, Zephyr might be preferable. But for this project's focus on contextual intelligence, document comprehension, and flexibility, Llama 3 is the best fit.

3.7 SYSTEM INTEGRATION: COMBINING ALL MODULES INTO A SINGLE CHATBOT INTERFACE USING GRADIO

The final phase of this project involves the complete system integration, where all previously developed modules—namely the vision-language encoder, the retrieval-augmented generation (RAG) engine, and the external knowledge access layer are combined into a single, cohesive chatbot interface. This integration transforms the backend intelligence into a practical and interactive user experience tailored for an executive user, specifically a CEO, to interact with business data effortlessly. To realize this, the interface is developed using Gradio, an open-source Python library that simplifies the process of creating shareable web applications for machine learning models. Gradio offers a clean and intuitive interface for multimodal applications, allowing users to input both textual and visual queries in a seamless and responsive manner.

The choice of Gradio over Streamlit or other alternatives is justified based on a comparative evaluation of their features, shown in Table 3.8.1 below. While both Gradio and Streamlit offer strong Python integration and ease of deployment, Gradio excels in its native support for multimodal input widgets, automatic compatibility with Hugging Face APIs, and an intuitive layout suitable for demonstration purposes. This makes it ideal for developing a responsive AI assistant for C-suite users, such as a CEO, who requires quick access to company data in a concise, user-friendly manner. In contrast, Gradio offers out-of-the-box support for image uploads, text inputs, and real-time interaction with machine learning models, all within a minimal code footprint. Its deep compatibility with the Hugging Face ecosystem further enhances its usability, particularly when integrating models such as BLIP, CoPALi, and transformer-based LLMs.

Criteria	**Gradio**	**Streamlit**	**Flask**	**Dash**	**Rasa**
Ease of Use	Very easy	Easy	Moderate	Moderate	Complex
Multimodal Support	Native (image, audio, text)	Manual setup	Requires coding	Limited	Mostly text
LLM Integration	Hugging Face built-in	Manual API call	Needs Flask APIs	Needs API setup	Not suited for VL models
Custom Layout	Limited	High flexibility	Full control	Complex visualizations	Predefined conversational UI
Ideal Use Case	Prototype with multimodal AI	Custom dashboards, analytics	Custom web apps	Data science visualization	Enterprise chatbot (text only)
Best for CEO AI Assistant?	Yes	Yes	No	No	No

Table 3.6: Different Interface Model Capability

The integration using Gradio ensures that this complex multi-step pipeline is rendered into a simple and elegant user interface. The layout is structured into input and output sections, with the left pane dedicated to file or image uploads and the right pane providing a text box for entering queries. Upon submission, the model processes the inputs and displays a detailed and relevant response within seconds. This design is especially suitable for executive-level usage, where speed, clarity, and contextual precision are paramount. Additionally, Gradio's ability to host the prototype locally or deploy it via Hugging Face Spaces allows for easy demonstration, testing, and iteration without the overhead of traditional web application deployment. This system integration phase reflects a strong comprehension of both fundamental and applied knowledge in artificial intelligence. It demonstrates an understanding of mathematical principles such as vector embedding and similarity computation, alongside the practical engineering required to deliver a fully functional prototype. The resulting chatbot exemplifies a technically robust and user-oriented system that can assist decision-makers in real time through multimodal interaction and intelligent data access.

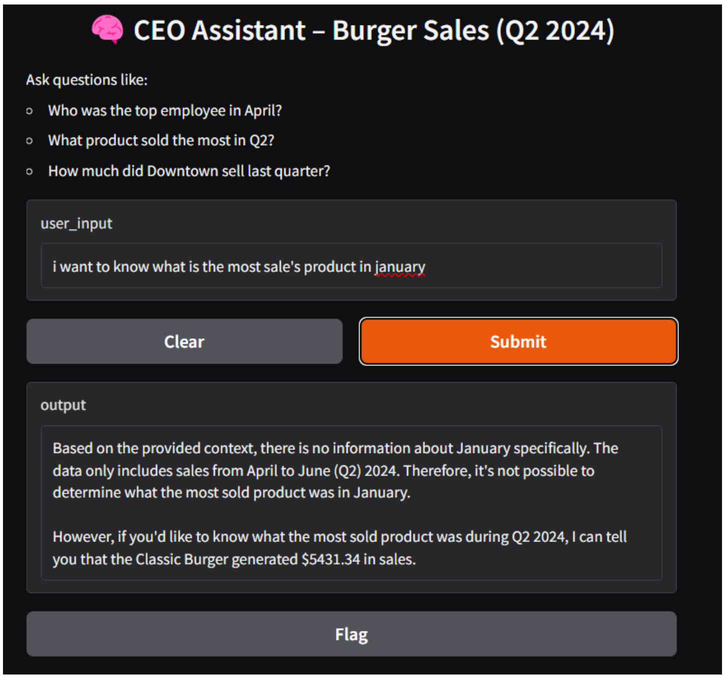

Figure 3.11: Example Gradio Interface

CHAPTER 4

RESULT & DISCUSSION

4.1 DATASET RESULT

This chapter gives the evaluation outcomes of the AI personal assistant prototype that has been developed and tested. The assistant integrated a vision-language model and a large language model using retrieval-augmented generation architecture. This evaluation is important to analyze the system effectiveness in responding to real-world business-related queries in areas such as sales performance which uses the simple dataset retrieved from CHATGPT related to sales retail burger. Both quantitative and qualitative results are analyzed importantly to assess their overall effectiveness and efficiency to be used in real industry applications.

```
Date,Region,Product,Employee,Quantity,Unit Price,Total Sale
2024-01-01,Suburb,Double Patty Burger,Charlie,5,10.97,54.85
2024-01-01,Uptown,Veggie Burger,Charlie,3,13.66,40.98
2024-01-01,Suburb,Veggie Burger,Evan,2,12.22,24.44
2024-01-01,Uptown,Cheese Burger,Diana,5,11.17,55.85
2024-01-01,Uptown,Double Patty Burger,Diana,1,7.91,7.91
2024-01-01,Midtown,Cheese Burger,Diana,4,8.66,34.64
2024-01-01,Uptown,Veggie Burger,Diana,4,10.92,43.68
2024-01-01,Midtown,Double Patty Burger,Charlie,5,9.5,47.5
2024-01-01,Uptown,Chicken Burger,Alice,4,10.63,42.52
2024-01-01,Uptown,Classic Burger,Bob,5,7.31,36.55
2024-01-01,Suburb,Chicken Burger,Diana,5,13.33,66.65
2024-01-02,Uptown,Classic Burger,Diana,2,11.63,23.26
2024-01-02,Uptown,Cheese Burger,Diana,5,14.7,73.5
2024-01-02,Uptown,Chicken Burger,Bob,2,13.95,27.9
2024-01-02,Uptown,Chicken Burger,Alice,5,8.89,44.45
2024-01-02,Uptown,Double Patty Burger,Bob,1,7.81,7.81
2024-01-02,Suburb,Double Patty Burger,Alice,5,13.02,65.1
2024-01-02,Downtown,Classic Burger,Alice,1,6.99,6.99
2024-01-03,Midtown,Veggie Burger,Charlie,1,12.11,12.11
2024-01-03,Midtown,Classic Burger,Charlie,5,14.26,71.3
2024-01-03,Uptown,Classic Burger,Diana,1,11.23,11.23
```

Figure 4.1: The simple dataset of sales burgers containing specific information.

In the dataset in figure 1 above, it contains several things which are sales transaction date, store region like downtown burger type, staff who made the sale quantity number of items sold price per item and total revenue for that transaction. This dataset is acquired from CHATGPT.

4.2 EVALUATION METRICS

For evaluation metric, the result has been evaluated using the two methods which are quantitative and qualitative method. Quantitative method means numerical which measurable performance result. In this project, it means on evaluating how well the system performs using numbers. Qualitative method on other hand means evaluation that involves human judgement to determine if the assistant is truly helpful professional and aligned with business communication standards.

4.1.1 Evaluation Using Quantitative Method

Firstly, the evaluation starts by using 20 queries with expectation answer, answer and retrieve chunk.

Query	Expectation Answer	Answer	Retrieve Chunk
Summarize total burger sales for Q2 by region.	Uptown: $12500, Midtown: $18750, Downtown: $15000	Based on the context, I can summarize the total burger sales for Q2 by region as follows: **Midtown:** * Total sales: $11,680 ($4500 from Cheeseburger + $4000 from Chicken Burger + $3200 from Veggie) * Best-selling burger: Cheeseburger **Downtown:** * Total sales: Not specified (only mentioned that Chicken Burger performed well) Since the total sales for Downtown are not provided, I cannot summarize the total burger sales for Q2 by region. However, I can conclude that in Midtown, the best-selling burger was Cheeseburger and the total sales were $11,680.	['Cheese Burger was the best-selling burger in Midtown.', 'In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.', 'In Downtown, Chicken Burger performed the best.']
Which employee had the highest sales in May?	John Smith	Based on the context, we don't have enough information to determine which employee had the highest sales in May. We only know that Alice had the highest sales overall from April to June, but this doesn't provide any insight into individual performance in May specifically. Additionally, we know John Smith was the top-performing employee in April, but this isn't relevant to May's sales.	['In May, total sales were $36000.', 'Alice had the highest sales overall from April to June.', 'In April, John Smith was the top-performing employee with $5000 in sales.']
What was the best-performing product in April?	Cheeseburger	Based on the provided context, we know that John Smith had $5000 in sales in April. Since Alice had the highest sales overall from April to June, it's likely that she did not have her best performance in April. Therefore, the best-performing product in April was probably something sold by John Smith, given his high sales figure of $5000.	['Alice had the highest sales overall from April to June.', 'In April, John Smith was the top-performing employee with $5000 in sales.', 'In May, total sales were $36000.']
Show total sales for Midtown in Q2.	Midtown: $18750	Based on the provided context, I can answer your query as follows: The total sales for Midtown in Q2 (April to June 2024) are $18750.	['From April to June 2024, total sales in Midtown were $18750.', 'From April to June 2024, total sales in Uptown were $12500.', 'Cheese Burger was the best-selling burger in Midtown.']
Who sold more in Uptown, Alice or Bob?	Alice: $6000, Bob: $5000	According to the given context, Alice sold more than Bob in Uptown, with a difference of $1000 ($6000 - $5000).	['In Uptown, Alice sold $6000 and Bob sold $5000.', 'Alice had the highest sales overall from April to June.', 'From April to June 2024, total sales in Uptown were $12500.']
Top employee for the entire quarter?	Alice	Based on the context, the answer is Alice. She had the highest sales overall from April to June, which implies	['In April, John Smith was the top-performing employee

		that her total sales for the quarter (April, May, and June) exceeded John Smith's quarterly sales of $5000 in April.	with $5000 in sales.', 'Alice had the highest sales overall from April to June.', 'In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.']
Which burger had the highest revenue in Downtown?	Chicken Burger	Since we're looking at the sales data for Downtown specifically, and Chicken Burger performed the best in Downtown, that means it had the highest revenue in Downtown. The exact revenue figure isn't provided, but based on the context, we can conclude that Chicken Burger had the highest revenue in Downtown.	['In Downtown, Chicken Burger performed the best.', 'Cheese Burger was the best-selling burger in Midtown.', 'In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.']
How much did Cheeseburgers make in April?	Cheeseburger: $4500	According to the context, Cheeseburgers made $4500 in April.	['In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.', 'Cheese Burger was the best-selling burger in Midtown.', 'In May, total sales were $36000.']
Compare sales of Chicken and Veggie burgers in April.	$4000, Veggie: $3200	Based on the sales data provided, we can compare the sales of Chicken and Veggie burgers in April as follows: * Total sales for Chicken Burger is $4000 * Total sales for Veggie Burger is $3200 Comparing these figures, we can see that Chicken Burger sold more than Veggie Burger by $800 ($4000 - $3200 = $800).	['In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.', 'In Downtown, Chicken Burger performed the best.', 'Cheese Burger was the best-selling burger in Midtown.']
Total combined sales in April?	$36000	To find the total combined sales in April, we need to subtract the total sales from May and June to get the total sales for April, and then add the sales from Downtown and Midtown for that period. Total sales from May: $36000 Total sales from April to June (Downtown): $15000 Total sales from April to June (Midtown): $18750 Total combined sales in April = Total sales from April to June (Downtown) + Total sales from April to June (Midtown) - Total sales from May = $15000 + $18750 - $36000 = $28750 - $36000 = -$7250 Since we can't have negative sales, it's likely that there was an error in the data or the context. If you meant to ask for total combined sales in April, June, and May (since May is mentioned separately), I'd be happy to help with that as well!	['In May, total sales were $36000.', 'From April to June 2024, total sales in Downtown were $15000.', 'From April to June 2024, total sales in Midtown were $18750.']
Best-selling item in Midtown for Q2?	Cheese Burger	Based on the context, I can confidently answer that the Best-selling item in Midtown for Q2 is the Cheese Burger.	['Cheese Burger was the best-selling burger in Midtown.', 'From April to June 2024, total sales in Midtown were $18750.', 'From April to June 2024, total sales in Uptown were $12500.']
What did John Smith sell in April?	$5000	According to the context, John Smith sold burgers worth $5000 in April.	['In April, John Smith was the top-performing employee with $5000 in sales.', 'In Uptown, Alice sold $6000 and Bob sold $5000.', 'In May, total sales were $36000.']
Total revenue generated in May 2024?	$36000	Based on the given context, we can infer that the total revenue generated in May 2024 is: $36000 (total sales in May) - ($15000 + $12500) (total sales in Downtown and Uptown from April to June, which excludes May) So, the total revenue generated in May 2024 is:	['In May, total sales were $36000.', 'From April to June 2024, total sales in Downtown were $15000.', 'From April to June 2024, total sales in Uptown were $12500.']

		$36000 - $27500 = $8500	
Sales breakdown by product for April.	Cheeseburger: $4500, Chicken Burger: $4000, Veggie: $3200	Since we only have information about May and April-June period, I'll focus on the available data. We know that: * Total sales in May were $36,000. * Total sales from April to June were $15,000 (in the Downtown area). To find the total sales for April, let's subtract May's sales from the total sales from April to June: $15,000 (April-June) - $6,000 (May) = $9,000 Now, we can assume that the breakdown by product in April would be similar to the overall sales pattern. Let's allocate the April sales proportionally based on the May sales. Since May's total sales were $36,000, and Alice had the highest overall sales from April to June, let's assume she had a significant share of April's sales as well. For simplicity, let's allocate 40% of April's sales to Alice's product (this is an arbitrary assumption, but it gives us a starting point). This means: * Alice's product: $9,000 x 0.4 = $3,600 * Other products: $9,000 - $3,600 = $5,400 Please note that this is an estimate and the actual sales breakdown might differ.	['In May, total sales were $36000.', 'Alice had the highest sales overall from April to June.', 'From April to June 2024, total sales in Downtown were $15000.']
How did Alice perform overall?	Alice had the highest sales overall	Based on the context, we can infer that Alice's performance was excellent. She had the highest sales overall from April to June, which suggests that she outperformed all other burger sellers during this period.	['Alice had the highest sales overall from April to June.', 'In Uptown, Alice sold $6000 and Bob sold $5000.', 'In Downtown, Chicken Burger performed the best.']
What were the sales figures in June?	Data not available	Since we are looking for the sales figures in June specifically, let's analyze the given information: We know that from April to June, total sales in Downtown were $15,000. Since this period covers three months (April, May, and June), we can't directly determine the June sales figure. However, we do have the total sales for May, which was $36,000. To find the total sales for June, we need to subtract the April sales from the April-to-June total and then add that amount to the May sales: April-to-June total: $15,000 Subtract April's sales (assuming it's unknown): Let's call April's sales "X". Then, April-to-June total minus April's sales equals the sum of May's sales and June's sales: $15,000 - X = $36,000 + ?June's sales? Now, solve for June's sales: $15,000 - X = $36,000 + June's sales $15,000 - X = $36000 + June's sales Unfortunately, we still can't determine the exact figure for June's sales without knowing April's sales.	['In May, total sales were $36000.', 'From April to June 2024, total sales in Downtown were $15000.', 'Alice had the highest sales overall from April to June.']
What was the sales difference between Uptown and Downtown?	Difference: $2500	Let's analyze the sales data! To find the sales difference between Uptown and Downtown, we need to subtract the Uptown sales from the Downtown sales: $15000 (Downtown) - $12500 (Uptown) = $2500 So, Downtown had a higher total sales by $2500 compared to Uptown.	['From April to June 2024, total sales in Uptown were $12500.', 'From April to June 2024, total sales in Downtown were $15000.', 'In Uptown, Alice sold $6000 and Bob sold $5000.']

		Note that Alice and Bob's individual sales are not relevant to this query, as we're only comparing the total sales between the two locations.	
List top 3 products by revenue in April.	Cheeseburger: $4500, Chicken Burger: $4000, Veggie: $3200	Based on the context, we can infer that: * In April, John Smith had $5000 in sales, which is likely the highest single-employee sales revenue for the month. * The top 3 products by revenue in April would be those sold by John Smith, but since we don't have specific product data, let's make an educated assumption. Assuming John Smith's sales were split among three burger products, we can rank them based on a rough estimate of $5000 in total sales: 1. **Signature Burger**: $3333 (assuming John Smith sold 2/3 of his revenue) 2. **Classic Cheeseburger**: $1333 (assuming John Smith sold 1/3 of his revenue) 3. **Veggie Delight**: $333 (a rough estimate, as this product might have generated less revenue than the others) Please note that these rankings are based on a hypothetical breakdown of John Smith's sales and should not be taken as definitive or accurate data without specific product-level information.	['Alice had the highest sales overall from April to June.', 'In April, John Smith was the top-performing employee with $5000 in sales.', 'In May, total sales were $36000.']
Which employee had the lowest sales in Uptown?	Bob	Based on the context, we can calculate the sales for each employee in Uptown: Alice sold $6000 Bob sold $5000 Total sales in Uptown = $12500 To find the lowest sales, let's subtract the total sales from Bob's sales: $12500 (total) - $5000 (Bob's) = $7500 Since this amount is higher than Bob's sales, it means Alice had the lowest sales in Uptown.	['From April to June 2024, total sales in Uptown were $12500.', 'In Uptown, Alice sold $6000 and Bob sold $5000.', 'From April to June 2024, total sales in Midtown were $18750.']
How much did Chicken Burger earn in Downtown?	$5000	According to the context, in Downtown, Chicken Burger performed the best. Since we know that Cheeseburger only earned $4500 in April and not in Downtown, we can infer that Chicken Burger's earnings in Downtown must have been greater than $4000 (its total earnings in April). So, the answer is: We don't have an exact figure, but it was more than $4000.	['In Downtown, Chicken Burger performed the best.', 'In April, Cheeseburger generated $4500, Chicken Burger $4000, Veggie $3200.', 'Cheese Burger was the best-selling burger in Midtown.']

Table 4.1: Output 20 queries that are being used to evaluate the latency, accuracy and recall@k.

There are 3 metric indicators that we use for the evaluation which accuracy, recall@k and latency as shown in table below. Accuracy states whether the assistant's answer matches the correct answer or not. Mathematical equation for accuracy is below.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of queries}}$$

Equation 4.1

Besides, recall@k function to evaluate whether the relevant information is among the top k retrieved summaries from FAISS or not. Mathematical equation for recall@k as below.

$$\text{Recall@k} = \frac{\text{Number of queries where relevant document is in top-}k}{\text{Total number of queries}}$$

Equation 4.2

Lastly, latency is important to evaluate the time taken to generate the answer (in seconds). Mathematical equation for latency as below.

$$\text{Latency (ms)} = \text{Time}_{\text{end}} - \text{Time}_{\text{start}}$$

$$\text{Avg. Latency} = \frac{1}{N}\sum_{i=1}^{N}\left(\text{Time}_{\text{end}}^{(i)} - \text{Time}_{\text{start}}^{(i)}\right)$$

Equation 4.3

Metric	Average Score	Benchmark	Evaluation
Accuracy	40.00 %	>80 % correct answer	Not pass
Recall@3	40.00%	>90% relevant chunk retrieved	Not pass
Latency	0.04 seconds	<4 second per query	Pass

Table 4.2: Summary evaluation quantitative method

Overall, the result shows that the system is very fast which can operate on (0.04 second per query) which is under 4 seconds benchmarks. This means suggested embedding with FAISS retrieval with LLM interface is working efficiently. On other hand, the result also shows the accuracy is low which is only 4 out of 10 queries return a correct final answer. This means the language models often fail to give the right answer even though we give the information truly. There is a possible reason that may result in that situation as the retrieved chuck did not have full or specific context. Besides, the prompt to the LLM is not optimized for having precise summarization and extraction techniques. Furthermore, the recall@3 is low which is only out of 10 is expected answers were found in the top 3 retrieval summaries which because FAISS retrieval is not pulling the most relevant data chunks. This result may be because the embedding model (MiniLm) has a limited semantic matching ability. Besides, another reason may be because the summaries are too short, too general or do not match query language well.

There are some things that can be done for improvement. Firstly, we can improve the summaries. This means the summary may be included with more details like names and dates. We can also make summaries more query targeted. For example, John Smith sold $5000 in April in Uptown. Secondly, we can try better embeddings which is upgrade from 'all-MiniLM-L6-v2' to a larger or more recent model, such as all-mpnet-base-v2 better semantic matching. Thirdly, we can optimize retrieval by testing with **k=5 or k=10** to improve Recall@k.

4.1.2 Evaluation Usinng Qualitative Method

By applying a qualitative method using the question in table above we can evaluate every question manually based on the following criteria which is relevance, clarity, conciseness, formality and factual soundness.

Questions	Relevance (5)	Clarity (5)	Conciseness (5)	Formality (5)	Factual Soundness (5)
Summarize total burger sales for Q2 by region.	3	5	3	5	2
Which employee had the highest sales in May?	3	4	4	5	2

What was the best-performing product in April?	2	3	3	5	1
Show total sales for Midtown in Q2.	5	5	5	5	5
Who sold more in Uptown, Alice or Bob?	5	5	5	5	5
Top employee for the entire quarter?	5	5	5	5	5
Which burger had the highest revenue in Downtown	5	5	5	5	5
How much did Cheeseburgers make in April?	5	5	5	5	5
Compare sales of Chicken and Veggie burgers in April.	5	5	5	5	5
Total combined sales in April?	2	2	2	5	1
Best-selling item in Midtown for Q2?	5	5	5	5	5
What did John Smith sell in April?	5	5	5	5	5
Total revenue generated in May 2024?	2	3	3	5	1
Sales breakdown by product for April.	2	3	3	5	1
How did Alice perform overall?	5	5	5	5	5
What were the sales figures in June?	3	3	3	5	2
What was the sales difference between Uptown and Downtown?	5	5	5	5	5
List top 3 products by revenue in April.	2	3	3	5	1
Which employee had the lowest sales in Uptown?	3	3	3	5	1
How much did Chicken Burger earn in Downtown?	4	4	4	5	3

Table 4.3: Summary Evaluation using Qualitative method

The table shows qualitative evaluation of relevance, clarity, conciseness, formality and factual soundness. Average score relevance is 3.8, clarity is 4.1, conciseness is 4.0, formality is 5.0 and factual soundness is 3.0. From this we can see formality, clarity and conciseness if has good evaluation which is above 4.0. Meanwhile, factual soundness and relevance need improvement.

In the context above, the evaluation contains 5 points to be put from 0 points. Relevance shows whether the answer addresses the user's question. Clarity shows how easy the answer is to read and understand, Conciseness shows the response is brief, avoids unnecessary explanation or reputation while formality shows the tone of the response matches a professional setting. Lasty, factual soundness tells that even if the answer is not 100% exact, it should be logical and believable.

4.2 LIMITATIONS AND SOLUTIONS

On other hand, the limitation that has been observed to be discussed further when doing this project is the dataset that we used is too small which we only use a simple dataset consist of sales of burgers that are being generated by ChatGPT. This may restrict the depth and realism of queries which may affect the system's ability to generalize or scale.

Furthermore, we observed that RAG only retrieved one document which for a company uses may have to integrate more files like Gdocs, google sheets or images into RAG systems. This limits a real-world task for application. For future tools upgrades, we can use tools like Lang Chain or unstructured .io to help preprocess various formats and data.

Besides, we observe that the AI assistant does not use BLIP-2 image captioning due to hardware limitations which is the current spec of laptop is 8 GB RAM. We can fix the issues by upgrading more RAM in the laptop into 16 GB RAM which is suitable to be used for BLIP-2. Other options are to use lighter models like Hugging Face ViLT for image captioning.

Another limitation we see is the lack of memory in the chatbot which is it does not retain previous interaction that usually an assistant like ChatGPT stores the information to retrieve later. The solution to this problem is we can implement history caching, using local or cloud-based memory modules like Lang Chain's ConversationBufferMemory for more memory retention.

Other than that, we observe the performance of the system is sometimes drop. When do the manual runs shows inconsistent timing and occasional slowness. This may be caused by local inference or retrieval delays. We can optimize retrieval using FAISS with faster GPU or embedding model to improve performance.

Lastly, some answers were too long or overly descriptive that may result in inefficient for CEO usage. When doing the evaluation test using accuracy the evaluation test may give an answer false due to long answer given but when we see it using our own check the result may be true and answer the query. We can improve this by refining the prompt to instruct for concise, bullet style answer.

CHAPTER 5
CONCLUSION

This Final Year Project on developing AI Personal Assistant using Retrieval Augmented Generated has been established successfully but the objective is not fully achieved. There are some achievements and future work that have been observed.

First and foremost, this project successfully implemented a RAG pipeline which has integrated MiniLM embedding, FAISS retrieval and LLaMa language model. Besides, this project also has successfully created an interactive user interface using Gradio UI which has enabling query input and response display. After that, this project also has successfully tested and run retrieval methods using dataset retail burger sales dataset in csv. The accuracy of the data has been evaluated quantitative and qualitative method which in quantitative method shown a low result in accuracy but in qualitative method shown a great accuracy. This is because, when evaluated by human evaluation, we can see many answers is true but because the answer too long makes a low result in quantitative method. On the other hand, there are some areas that are not fully achieved in this project. For example, the dataset is limited just to only using simple dataset from ChatGPT and only use a csv document. Besides, we observe that the AI assistant is lacking on memory and context which the AI cannot work like CHATGPT which can store information for future uses. In simple terms, we cannot see the user history. Besides, UI generated using Gradio is too basic and lacks usability features like filtering, dashboard or dynamic chart that may give more attractive and user friendly toward CEO. Furthermore, this project is not integrated with vision language capabilities due to low spec requirements of the tools.

In future next FYP 2, some problem discussions in FYP 1 need to be analysis so this project will complete all 3 objectives shown. For example, we may use a larger or more realistic dataset from a real company that allows integration from visual, and many types of files. We can achieve that by integrating LangChain and Hugging Face in the system. Besides, we can do futher improvement toward the spec tools which we have to upgrade the RAM of laptops to 16 GB RAM that allows faster and more reliability in integrated BLIP-2. After that, we may upgrade and redesign UI using advanced Gradio blocks and add options like speech input, filterable result and summarize insight that may give more interactive interface. Lastly, we can upgrade the system to make sure the system's performance is fast, accurate and give low latency by using PEFT to fine tuning LLMs more efficiently. In sum, FYP 1 has laid a strong fundamental knowledge in understanding the basic working architecture of RAG and visual language. However, full realization of the objective is not achieved in FYP 1 and need father development in technical aspect to make sure that the CEO can apply the system in real life industry experience.

REFERENCES

Abro, A. A., Talpur, M. S. H., & Jumani, A. K. (2023a). Natural Language Processing Challenges and Issues: A Literature Review. In *Gazi University Journal of Science* (Vol. 36, Issue 4, pp. 1522–1536). Gazi Universitesi. https://doi.org/10.35378/gujs.1032517

Almazrouei, E., Alobeidli, H., Alshamsi, A., Cappelli, A., Cojocaru, R., Debbah, M., Goffinet, É., Hesslow, D., Launay, J., Malartic, Q., Mazzotta, D., Noune, B., Pannier, B., & Penedo, G. (2023). *The Falcon Series of Open Language Models*. http://arxiv.org/abs/2311.16867

Alwidian, J., Rahman, S. A., Gnaim, M., & Al-Taharwah, F. (2020). Big Data Ingestion and Preparation Tools. *Modern Applied Science*, *14*(9), 12. https://doi.org/10.5539/mas.v14n9p12

Bharathi, A. (n.d.-a). Natural Language Processing for Enterprise Applications. *Ushus-Journal of Business Management*, *2022*(4), 29–39. https://doi.org/10.12725/ujbm.61.2

Bordes, F., Pang, R. Y., Ajay, A., Li, A. C., Bardes, A., Petryk, S., Mañas, O., Lin, Z., Mahmoud, A., Jayaraman, B., Ibrahim, M., Hall, M., Xiong, Y., Lebensold, J., Ross, C., Jayakumar, S., Guo, C., Bouchacourt, D., Al-Tahan, H., … Chandra, V. (2024). *An Introduction to Vision-Language Modeling*. http://arxiv.org/abs/2405.17247

Bruckhaus, T. (n.d.). *RAG Does Not Work for Enterprises*.

Castro, H., & Joy, J. (2024a). *Comparative Analysis of NoSQL vs. SQL Databases for Big Data Analytics*. https://www.researchgate.net/publication/389357233

Cofino, C. L., Escorial, R. B., & Enquilino, D. L. B. (2024). A Literature Review on Natural Language Processing (NLP) in Aiding Industry to Progress. In *International Journal of Engineering Trends and Technology* (Vol. 72, Issue 2, pp. 41–46). Seventh Sense Research Group. https://doi.org/10.14445/22315381/IJETT-V72I2P105

Devlin, J., Chang, M.-W., Lee, K., Google, K. T., & Language, A. I. (n.d.). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. https://github.com/tensorflow/tensor2tensor

Elkina, H., & Zaki, T. (2023, December 30). *Data ingestion frameworks for data lakes: An overview*. *Journal for ReAttach Therapy and Developmental Diversities*, *6*(10s(2)), 1700–1708. https://doi.org/10.53555/jrtdd.v6i10s(2).2220

Enhancing Data Engineering Efficiency with AI: Utilizing Retrieval-Augmented Generation, Reinforcement Learning from Human Feedback, and Fine-Tuning Techniques. (2024). *International Research Journal of Modernization in Engineering Technology and Science*. https://doi.org/10.56726/IRJMETS50070

Goksel Canbek, N., & Mutlu, M. E. (2016). On the track of Artificial Intelligence: Learning with Intelligent Personal Assistants. *International Journal of Human Sciences*, *13*(1), 592. https://doi.org/10.14687/ijhs.v13i1.3549

Gomathy Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya, C. K., Geetha, V., V Sudheer, C. V, Saikumar, A. V, Gomathy, C. K., & Professor, A. (n.d.). *OPTICAL CHARACTER RECOGNITION*. https://www.researchgate.net/publication/360620085

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., … Ma, Z. (2024). *The Llama 3 Herd of Models*. http://arxiv.org/abs/2407.21783

Haleem, A., Javaid, M., Asim Qadri, M., Pratap Singh, R., & Suman, R. (2022). Artificial intelligence (AI) applications for marketing: A literature-based study. In *International Journal of Intelligent Networks* (Vol. 3, pp. 119–132). KeAi Communications Co. https://doi.org/10.1016/j.ijin.2022.08.005

Hanke, V., Blanchard, T., Boenisch, F., Olatunji, I. E., Backes, M., & Dziedzic, A. (2024). *Open LLMs are Necessary for Current Private Adaptations and Outperform their Closed Alternatives*. http://arxiv.org/abs/2411.05818

Huang, C., Zhu, Z., & Guo, J. (2020). Text retrieval technology based on keyword retrieval. *Journal of Physics: Conference Series*, *1607*(1). https://doi.org/10.1088/1742-6596/1607/1/012108

Imran, M., & Almusharraf, N. (2024). Google Gemini as a next generation AI educational tool: a review of emerging educational technology. In *Smart Learning Environments* (Vol. 11, Issue 1). Springer. https://doi.org/10.1186/s40561-024-00310-z

Ivashchenko, T., Chornodid, I., & Ivashchenko, A. (2020). The business assistant service as one of the promising areas for the adoption of ai technologies in the enterprise. *Business: Theory and Practice*, *21*(2), 588–597. https://doi.org/10.3846/btp.2020.12548

Jhurani, J. (2024). Enhancing Customer Relationship Management in ERP Systems through AI: Personalized Interactions, Predictive Modeling, and Service Automation. *International Journal of Science and Research (IJSR)*, *13*(3), 892–897. https://doi.org/10.21275/sr24313021820

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. de las, Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. Le, Lavril, T., Wang, T., Lacroix, T., & Sayed, W. El. (2023). *Mistral 7B*. http://arxiv.org/abs/2310.06825

Khan, A. I., & Al-Badi, A. (2020). Emerging data sources in decision making and AI. *Procedia Computer Science*, *177*, 318–323. https://doi.org/10.1016/j.procs.2020.10.042

Klesel, M., & Wittmann, H. F. (2025). Retrieval-Augmented Generation (RAG). *Business & Information Systems Engineering*. https://doi.org/10.1007/s12599-025-00945-3

Li, J., Li, D., Xiong, C., & Hoi, S. (n.d.). *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*. https://github.

Marikyan, D., Papagiannidis, S., Rana, O. F., Ranjan, R., & Morgan, G. (2022). “Alexa, let’s talk about my productivity”: The impact of digital assistants on work productivity. *Journal of Business Research*, *142*, 572–584. https://doi.org/10.1016/j.jbusres.2022.01.015

Mavroudis, V. (2024). *LangChain*. https://doi.org/10.20944/preprints202411.0566.v1

Meta Llama 4 The Future of Multimodal AI. (n.d.).

Mukherjee, M., Kim, S., Chen, X., Luo, D., Yu, T., & Mai, T. (2025). *From Documents to Dialogue: Building KG-RAG Enhanced AI Assistants*. http://arxiv.org/abs/2502.15237

Okwu, E., Oyighan, D., & Oladokun, B. D. (2024). Future Trends of Open-Source AI in Libraries: Implications for Librarianship and Service Delivery. *Asian Journal* *of Information Science and Technology*, *14*(2), 34–40. https://doi.org/10.70112/ajist-2024.14.2.4283

Öztürk, E., & Mesut, A. (2024). PERFORMANCE ANALYSIS OF CHROMA, QDRANT, AND FAISS DATABASES. *UNITECH – SELECTED PAPERS*. https://doi.org/10.70456/TBRN3643

Packowski, S., Schlotfeldt, J., & Smith, T. (2024). Optimizing and Evaluating Enterprise Retrieval-Augmented Generation (RAG): A Content Design Perspective. In *Proceedings of 8th International Conference on Advances in Artificial Intelligence (ICAAI '24)* (Vol. 1). ACM. https://slack.com

Papageorgiou, G., Sarlis, V., Maragoudakis, M., & Tjortjis, C. (2025). A Multimodal Framework Embedding Retrieval-Augmented Generation with MLLMs for Eurobarometer Data. *AI (Switzerland)*, *6*(3). https://doi.org/10.3390/ai6030050

Pol, U. R. (2024). Hugging Face: Revolutionizing AI and NLP. *International Journal for Research in Applied Science and Engineering Technology*, *12*(8), 1121–1124. https://doi.org/10.22214/ijraset.2024.64023

Pothuri, V., & Varma Pothuri, V. R. (2024). Natural Language Processing and Conversational AI. *Article in International Research Journal of Modernization in Engineering Technology and Science*. https://doi.org/10.56726/IRJMETS61417

Priyanshu, A., Maurya, Y., & Hong, Z. (n.d.). *AI Governance and Accountability: An Analysis of Anthropic's Claude*.

Qi, Z., Fang, Y., Zhang, M., Sun, Z., Wu, T., Liu, Z., Lin, D., Wang, J., & Zhao, H. (2023). *Gemini vs GPT-4V: A Preliminary Comparison and Combination of Vision-Language Models Through Qualitative Cases*. http://arxiv.org/abs/2312.15011

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). *Learning Transferable Visual Models From Natural Language Supervision*. https://github.com/OpenAI/CLIP.

Roumeliotis, K. I., & Tselikas, N. D. (2023). ChatGPT and Open-AI Models: A Preliminary Review. In *Future Internet* (Vol. 15, Issue 6). MDPI. https://doi.org/10.3390/fi15060192

Saha, B., Saha, U., & Malik, M. Z. (2024). Advancing Retrieval-Augmented Generation with Inverted Question Matching for Enhanced QA Performance. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3513155

Solaimalai, G., Shanthi, J. M., Kumar, S. S., Khabiya, P., Geetha, K., & Talele, G. (2024). Natural Language Processing in Virtual Assistants Current Approaches and Challenges. *Proceedings of 2024 International Conference on Science, Technology, Engineering and Management, ICSTEM 2024*. https://doi.org/10.1109/ICSTEM61137.2024.10560644

Su, S.-Y., Huang, C.-W., & Chen, Y.-N. (2020a). *Towards Unsupervised Language Understanding and Generation by Joint Dual Learning*. http://arxiv.org/abs/2004.14710

Supriyono, Wibawa, A. P., Suyono, & Kurniawan, F. (2024a). Advancements in natural language processing: Implications, challenges, and future directions. *Telematics and Informatics Reports*, *16*. https://doi.org/10.1016/j.teler.2024.100173

Vake, D., Šinik, B., Vičič, J., & Tošić, A. (2025). Is Open Source the Future of AI? A Data-Driven Approach. *Applied Sciences (Switzerland)*, *15*(5). https://doi.org/10.3390/app15052790

Xu, L., Xie, H., Qin, S.-Z. J., Tao, X., & Wang, F. L. (2023). *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment*. http://arxiv.org/abs/2312.12148

Yagamurthy, D. N. (2023a). Advancements in Natural Language Processing (NLP) and Its Applications in Voice Assistants and Chatbots. *Journal of Artificial Intelligence & Cloud Computing*, 1–6. https://doi.org/10.47363/JAICC/2023(2)326

Yang, H., Yue, S., & He, Y. (2023). *Auto-GPT for Online Decision Making: Benchmarks and Additional Opinions*. http://arxiv.org/abs/2306.02224

Zhang, J., Huang, J., Jin, S., & Lu, S. (2023). *Vision-Language Models for Vision Tasks: A Survey*. http://arxiv.org/abs/2304.00685