

Multimodal RAG Analysis of Product Datasheet

David Lau Keat Jin¹, Ganthan Narayana Samy², Fiza Abdul Rahim³, Nurazean Maarop⁴, Mahiswaran Selvananthan⁵, Mazlan Ali⁶ & Sundresan Perumal⁷

^{1,2,3,4}*Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, Kuala Lumpur, 54100, Malaysia*

^{5,6}*Faculty of Social Sciences and Humanity, Universiti Teknologi Malaysia, Kuala Lumpur, 54100, Malaysia*

⁷*Faculty of Science and Technology, Universiti Sains Islam Nilai, 71800, Malaysia*
davidkeat@graduate.utm.my

Article history

Received:
25 Oct 2024

Received in revised form:
18 Nov 2024

Accepted:
30 Nov 2024

Published online:
27 Dec 2024

*Corresponding author
davidkeat@graduate.utm.my

Abstract

Large language models such as ChatGPT serves as multipurpose chatbot that can provide information across diverse disciplines. However, in order to generate timely and accurate response, retrieval-augmented generation method has been devised to enhance the response of these models. The release of vision models has paved the way for practitioners to perform multimodal retrieval augmented generation on documents that commonly consist of a combination of text, images and tables. Hence, this method is explored to analyze a product datasheet and match it with minimum specification required by potential clients. It is demonstrated that multimodal retrieval augmented generation performed better compared to basic retrieval augmented generation that did not consider the information contained in image and table specifically. While the performance of this method still lagged behind the commercially available GPT-4o, information is not exchanged with any external parties which could potentially address any privacy issue with regards to highly sensitive information. The incorporation of various best practices in this domain as highlighted in other studies can potentially improve the output generation of this method toward matching or exceeding the performance of commercially available tools.

Keywords: Retrieval-augmented Generation, Multimodal RAG, Vision Model

1. Introduction

The advancement of Large Language Models (LLMs) has fundamentally transformed the domain of Natural Language Processing (NLP) and Artificial Intelligence (AI) [1]. The NLP capability of LLM leapfrogged after the discovery of transformer architecture, thus spurred the race for development of various LLM by large corporations after the year 2017 [2]. ChatGPT is one of the many of LLMs which was made possible by the discovery of transformer mechanism by Google researchers in their seminal paper "Attention Is All You Need". It has brought about a paradigm shift in the field of NLP by enabling efficient processing of sequential data, thereby facilitating parallelization, and capturing long-range dependencies in textual information. From foundational architectures such as OpenAI's GPT series

* Corresponding author. davidkeat@graduate.utm.my

to their open-source equivalents like Google's BERT and Facebook's RoBERTa, the emergence of both proprietary and publicly available models has facilitated a democratization of access to advanced language comprehension and generation capabilities [3]. These progressions have not only improved the efficacy of language-related functions but have also catalyzed innovation across a wide array of applications, encompassing automated customer support to sophisticated content creation and further applications. For example, it can be used to provide medical advice and diagnostic for medical practitioners in natural language [4]. Due to the tendency for LLM to hallucinate and the requirement to supplement LLM with internal information of an organization in a chatbot application, the retrieval-augmented-generation (RAG) was introduced [5]. The process of constructing a basic RAG system is illustrated in Figure 1 and consists of the following steps:

1. Identifying the internal documents required and store them in a single repository.
2. The documents are divided into chunks of text.
3. The chunks are embedded and place in vector store.
4. For a chatbot application, the query is normally obtained from the users via a system prompt which would trigger a search in the vector store.
5. The vector store will retrieve and produce relevant text according to the user input.
6. Both the user input and relevant text will be combined as prompt to the Language Model.
7. The Language Model will produce the result as the final output.

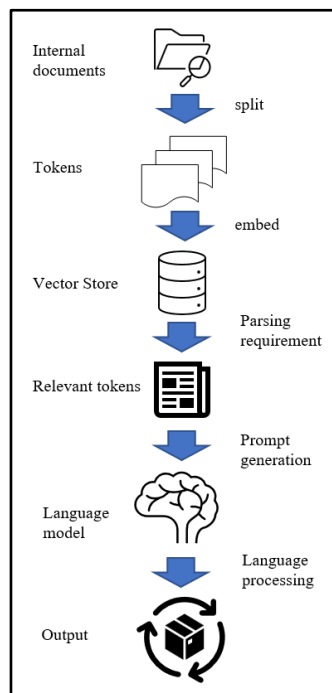


Figure 1. Basic RAG Process for Document-based Inference

Despite the encouraging developments, the incorporation RAG methodologies within LLMs poses numerous obstacles, particularly regarding the interpretation of

non-textual elements within documents [6]. RAG encompasses the enhancement of generative frameworks through the application of external retrieval systems aimed at improving the quality and pertinence of generated outputs by utilizing extensive information repositories [7]. Nevertheless, the effective retrieval and analysis of non-textual data—such as diagrams, charts, and images—demands advanced mechanisms to reconcile the semantic disparities between various data modalities. This intricacy is further exacerbated by the intrinsic heterogeneity and vagueness inherent in non-textual content, which may impede the precision and dependability of the retrieval operations. Tackling these issues is imperative for the fluid integration of multimodal data into generative models, thereby ensuring that LLMs can exploit the complete range of available information to generate coherent and contextually relevant outputs.

In light of these challenges, the advent of vision models, embeddings and its associated vector storage systems are promising progress. Visual models, including Convolutional Neural Networks (CNNs) and transformer-based frameworks such as Vision Transformers (ViTs), demonstrate superior proficiency in extracting salient features from images and other forms of visual data [8]. Through the generation of high-dimensional embeddings that encapsulate the quintessence of visual content, these models enable the alignment and amalgamation of visual and textual information within a cohesive paradigm. Vector storage systems, which adeptly index and retrieve these embeddings, serve a crucial function in facilitating scalable and efficient retrieval operations that are indispensable for RAG systems. The interplay between sophisticated vision models and resilient vector storage solutions empowers LLMs to adeptly incorporate and leverage visual information, thereby augmenting their multimodal understanding and generative capabilities [9].

Thus, a crucial advancement of LLM is the integration of multimodality—defined as the capability of models to process and synthesize information across various data modalities, encompassing text, images, audio, and video [10]. In this regard, multimodal language models expand the functionalities of conventional LLMs by facilitating more exhaustive and contextually nuanced interactions. For example, in instances where textual information is inadequate, the incorporation of visual data can augment the context, resulting in a more precise and sophisticated understanding and generation. The application of multimodality in language models enables functionalities such as image captioning, visual question answering, and enhanced content synthesis, consequently expanding the breadth and efficacy of AI-driven solutions. The pervasiveness of chatbots like ChatGPT which was introduced in November 2022 was due to its ability to generate answers across a diverse fields of knowledge [11]. It is no wonder that a relative majority (51 out of 230) of use cases reported in European public sector are in applications of chatbots, intelligent digital assistants, virtual agents and recommendation systems [12]. Notwithstanding the versatility of LLM, its most notable weakness is its tendency to hallucinate, which is to respond with erroneous answers confidently [13]. This is due to the limitations on the available data from which an LLM was trained upon prior to its cut-off date.

In conjunction with these technological progressions, platforms such as Ollama that endorse the deployment and utilization of a myriad of open-source models have emerged [14]. This platform serves as a paradigm of the movement towards

accessible and adaptable AI infrastructures, empowering researchers and practitioners to harness a diverse spectrum of language and vision models devoid of the limitations typically linked with proprietary systems such as rate-limit and monthly quota. By cultivating an environment that is conducive to experimentation and innovation, Ollama and analogous platforms that facilitate the operation of models on adequately equipped machines promote the exploration of novel architectures and integration methodologies, thereby expediting the development and adoption of advanced multimodal LLMs. The existence of such platforms democratizes access to state-of-the-art AI tools, fostering collaborative advancements and the proliferation of open-source initiatives that propel the field forward.

While research is still on-going regarding the effective methods in extraction, storage and analysis of multimodal information that are commonly available in research papers, company profiles, product brochures and datasheets, this study explored the use of models and storage systems to extract, store and analyze the three forms of information which include text, image and table. Currently, there are proprietary as well as open-source solutions that integrate with Python programming language in the manipulation of different modalities. For organizations that need to use highly-sensitive documents in RAG or have concern regarding the rate-limit imposed by paid subscription associated with online services, this study serves as a useful guide for all-local implementation for a multimodal RAG application.

2. Method

The program should ingest PDF documents with three different modalities: text, table and image. This may be internal document associated with an organization. In this setup, a product datasheet is used. The information extracted was stored in both vector store and document store where a common ID linked the related piece of information. To use the program, a user interface is built for user to upload an excel file with a column of attribute that need to be searched, in this case, it should be named as 'Minimum Specification'. The information to be searched was extracted and parsed to the inference model row-by-row and the equivalent response was generated. Upon the completion of the process, two new columns called 'Multimodal RAG' and 'Basic RAG' were generated and populated with the results of inference next to the column of information extracted. The algorithm for the program is provided in Table 1.

Table 1. Algorithm for the Program

<p>Input:</p> <ul style="list-style-type: none"> • PDF_Document: A sample PDF file which in this case a server datasheet. • Excel_File: An Excel file with a column named 'Minimum Specification'. <p>Output: Updated_Excel_File: Excel file with two added columns: 'Multimodal RAG' and 'Basic RAG'.</p> <p>Begin Algorithm</p> <p>Step 1: Initialize Data Stores Initialize VectorStore Initialize DocumentStore</p> <p>Step 2: Ingest and Process PDF_Document</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; padding: 5px;"><u>Multimodal RAG</u></th> <th style="text-align: center; padding: 5px;"><u>Basic RAG</u></th> </tr> </thead> <tbody> <tr> <td style="padding: 5px; vertical-align: top;"> <p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Extract text chunks – Extract tables – Extract images – Assign Unique Document ID – Store extracted text chunks, table and images in redis server (document store) – Generate summaries for text chunks, tables and images – Generate embeddings for the summaries – Store summaries with embeddings in chroma database (vector store) with similar ID to document store <p>End For</p> </td> <td style="padding: 5px; vertical-align: top;"> <p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Load the document – Segmentize it into chunks – Generate embedding for each chunk – Store the embedding and chunk into vector store <p>End For</p> </td> </tr> </tbody> </table>		<u>Multimodal RAG</u>	<u>Basic RAG</u>	<p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Extract text chunks – Extract tables – Extract images – Assign Unique Document ID – Store extracted text chunks, table and images in redis server (document store) – Generate summaries for text chunks, tables and images – Generate embeddings for the summaries – Store summaries with embeddings in chroma database (vector store) with similar ID to document store <p>End For</p>	<p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Load the document – Segmentize it into chunks – Generate embedding for each chunk – Store the embedding and chunk into vector store <p>End For</p>
<u>Multimodal RAG</u>	<u>Basic RAG</u>				
<p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Extract text chunks – Extract tables – Extract images – Assign Unique Document ID – Store extracted text chunks, table and images in redis server (document store) – Generate summaries for text chunks, tables and images – Generate embeddings for the summaries – Store summaries with embeddings in chroma database (vector store) with similar ID to document store <p>End For</p>	<p>For each PDF_Document in Datasheet folder do</p> <ul style="list-style-type: none"> – Load the document – Segmentize it into chunks – Generate embedding for each chunk – Store the embedding and chunk into vector store <p>End For</p>				
<p>Step 3: Generate inference from Excel_File</p> <p>For each Row in 'Minimum Specification' column do</p> <ul style="list-style-type: none"> – Extract the input – Parse the input to multimodal_rag – Enter the result into 'Multimodal RAG' column – Parse the input to basic_rag – Enter the result into 'Basic RAG' column <p>End For</p>					

- Save the new file as Updated_Excel_File

End Algorithm

For development and testing purposes, a virtual environment was created using Anaconda Navigator. In this virtual environment, Python was installed and used as the programming language. Figure 2 illustrates the information flow in this environment where Visual Studio is used as the code editor. The full list of software packages used is given in the file requirements.txt on github (<https://github.com/renaissance2005/dual-rag-comparison/>). In addition, redis server and ollama were installed on the machine. Subsequently, the following models were downloaded and installed using Ollama:

1. llama3.1:8b [15]
2. llava-llama3 [16]
3. nomic-embed-text [17]

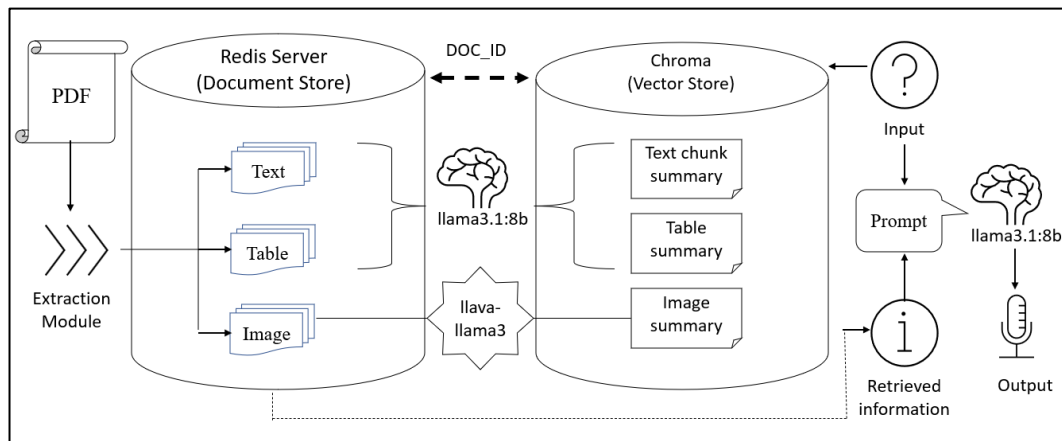


Figure 2. Information Flow for Development Environment

These three models are all available in Ollama platform and can be run on a local machine. llama3.1:8b is a text-based, open source LLM developed and released by Meta. Though it has the larger 70b model which can produce better inference, the 8b model is chosen as it is sufficient to achieve the objective established for this study. In addition, llava-llama3 is a vision model based on llava that has been fine-tuned with Llama 3 Instruct and CLIP-ViT-Large-patch14-336 with ShareGPT4V-PT and InternVL-SFT. Other than RAG, fine-tuning is a mechanism to improve performance of models in domain-specific inference [18]. Prior to storing the data into vector store like Chroma, generation of vector embeddings for the data is required. Hence, nomic-embed-text is chosen as it can embed both image and text. Finally, to benchmark the results from both multimodal and basic RAGs, the PDF document will be uploaded to the widely used and commercially available tool called GPT-4o (<https://chatgpt.com/>). The response from GPT-4o will be entered into another column named 'GPT Output' for comparison. The PDF document used is attached as Appendix 1.

3. Result

The output generated by both multimodal RAG and basic RAG are given in Table 2 where incorrect output is denoted with 'X' and incomplete output is denoted with '?'. In addition, the output generated by GPT-4o is also depicted for comparison. Based on the result, the multimodal RAG correctly responded for 8 out of 10 inputs in the form of minimum specification. Notably, the output for item 9 is incomplete. On the other hand, the basic RAG correctly generated 6 out of 10 outputs given the same input from the minimum specification. For benchmarking purpose, the similar instruction given to multimodal RAG and basic RAG for generation of the required output is applied for GPT-4o. The only difference is that the instruction as input is given to this online model sequentially from item 1 until 10 from 'Minimum Specification' column with the same PDF document uploaded each time. The instruction given is:

“You are an pre-sales specialist well-versed with the company products based on the datasheet and brochure available and are competent in matching the requirements from potential users or clients with the available product specification. You are given a PDF document which will be a mix of text, tables, and images usually of charts or graphs. Use this information to match the specification for provided Minimum specification with the relevant specification offered in the PDF document. Your answer should specify a value or text extracted as a piece of information from the PDF document only. DO NOT include preamble. DO NOT give me the sources where you obtain your information or conclusion. DO NOT ask further questions. DO NOT make up answers, use the provided context documents below and answer the question to the best of your ability. Be as brief, complete and precise in your output as possible.

#Example:

Minimum specification: 'Weight: Maximum 3kg'



Output: 'Minimum weight starts at 1.21kg'


Minimum specification given is {input from 'Minimum Specification' column}. All information extracted must be based on the PDF document submitted here.”

Table 2. Result of Analysis

No	Minimum Specification	Expected Response	Multimodal RAG	Basic RAG	GPT Output
1	Processor type: Intel	2nd Generation Intel® Xeon® Scalable processors	Intel	Up to two 2nd Generation Intel Xeon Scalable processors, up to 28 cores per processor	2nd Generation Intel® Xeon® Scalable processors

No	Minimum Specification	Expected Response	Multimodal RAG	Basic RAG	GPT Output
2	Processor core available: 12 core	up to 28 cores per processor	Up to 28 cores per processor	192 GB Max ✗	Up to 28 cores per processor
3	Processor speed: 3.6GHz	Not mentioned	Up to 28 cores per processor ✗	192 GB Max ✗	Up to two 2nd Generation Intel® Xeon® Scalable processors, up to 28 cores per processor. ✗
4	Storage Controllers: Just state what is offered	Internal Controllers: PERC H330, H730P, H740P, HBA330 External Controllers: H840, 12 Gbps SAS HBA Software RAID:S140	Storage Controllers: Just-in-Time (JIT) storage provisioning and System Erase for quick and secure data erasure are offered to increase database performance by up to 10x with Persistent Memory NVDIMM-N.	Up to 12 NVDIMM, 192 GB Max and up to 6 Intel Optane DC persistent memory PMem, 6.14TB max (7.68TB max with PMem + LRDIMM)	Internal Controllers : PERC H330, H730P, H740P, HBA330 External Controllers : H840, 12 Gbps SAS HBA Software RAID: S140 Boot Optimized Storage Subsystem (BOSS): HWRAID 2 x M.2 SSDs 240GB, 480GB Internal Dual SD Module

No	Minimum Specification	Expected Response	Multimodal RAG	Basic RAG	GPT Output
5	Storage: Up to 128TB	Front drive bays: Up to 16 x 2.5" SAS/SATA (HDD/SSD) max 122.88TB or up to 8 x 3.5" SAS/SATA HDD max 128TB	Up to 128TB	128TB	Up to 128TB with 8 x 3.5" SAS/SATA HDD
6	Memory Slots: at least 12 DDR4 slots	24 DDR4 DIMM slots, Supports RDIMM /LRDIMM,	Up to 12 NVDIMM	6.14TB max 	24 DDR4 DIMM slots
7	Input/Output & Ports	Network daughter card options 4 x 1GbE or 2 x 10GbE + 2 x 1GbE or 4 x 10GbE or 2 x 25GbE Front ports: 1 x Dedicated iDRAC Direct Micro-USB, 2 x USB 2.0, 1 x USB 3.0 (optional), 1 x VGA Rear ports: 1 x Dedicated iDRAC network port, 1 x	Based on the provided context documents, the closest specification that could meet or exceed the technical requirement of Input/Output & Ports is: * Up to 16 2.5" drives or 8 3.5" drives for storage scalability, allowing for flexibility and adaptability in application deployments. * Support for up to three double-width	Up to 16 2.5" drives or 8 3.5" drives. 	Network daughter card options: 4 x 1GbE or 2 x 10GbE + 2 x 1GbE or 4 x 10GbE or 2 x 25GbE Front ports: 1 x Dedicated iDRAC Direct Micro-USB, 2 x USB 2.0, 1 x USB 3.0 (optional), 1 x VGA Rear ports: 1 x Dedicated iDRAC network port, 1 x Serial, 2 x

No	Minimum Specification	Expected Response	Multimodal RAG	Basic RAG	GPT Output
		Serial, 2 x USB 3.0, 1 x VGA Video card: 2 x VGA Riser options with up to 8 PCIe Gen 3 slots, maximum of 4 x 16 slots	or four single-width FPGAs, enabling scalable business architecture for workload acceleration. This specification is mentioned in the PowerEdge R740 spec sheet as a feature that provides versatility to adapt to virtually any application.		USB 3.0, 1 x VGA Video card: 2 x VGA Riser options with up to 8 PCIe Gen 3 slots, maximum of 4 x 16 slots
8	Accelerator options	Up to three 300W or six 150W GPUs, or up to three double-width or four single-width FPGAs.	Up to three 300W or six 150W GPUs, See Dell.com/GPU for latest information.	Up to three 300W or six 150W GPUs	Up to three 300W or six 150W GPUs, or up to three double-width or four single-width FPGAs.
9	Supported operating systems	Canonical® Ubuntu® Server LTS Citrix® Hypervisor Microsoft Windows Server® LTSC with Hyper-V Oracle® Linux	Canonical Ubuntu Server LTS 	Canonical® Ubuntu Server LTS, Citrix Hypervisor, Microsoft Windows Server LTSC with Hyper-V, Oracle Linux, Red Hat	Canonical® Ubuntu® Server LTS, Citrix® Hypervisor, Microsoft Windows Server® LTSC with Hyper-V, Oracle®

No	Minimum Specification	Expected Response	Multimodal RAG	Basic RAG	GPT Output
				Enterprise Linux, SUSE Linux Enterprise Server, VMware ESXi.	Linux, Red Hat® Enterprise Linux, SUSE® Linux Enterprise Server, VMware® ESXi
10	Dimensions	Form factor: Rack (2U)	Dimensions Up to 16 x 2.5” SAS/SATA (HDD/SSD) max 122.88TB or up to 8 x 3.5” SAS/SATA HDD max 128TB	Width3: 434.0mm (17.08”) Depth3: 737.5mm (29.03”)	Form factor: Rack (2U) Height: 86.8mm (3.4”) Width: 434.0mm (17.08”) Depth: 737.5mm (29.03”) Weight: 28.6kg (63lbs.)

All the three methods returned information pertaining to processor core for item 3 when processor speed is required. Specifically, this information is not mentioned in the provided document. While multimodal RAG produced the incorrect information for item 10 which require the product dimensions, the provided output is actually the dimensions of the storage that comes with the server, which is semantically correct. Overall, the basic RAG performed poorest with incorrect analysis for item 2, 3, 6 and 7. While GPT-4o performed best, it only provided output for first 5 items if the same excel file is uploaded with the same prompt, indicating possible limitation on rate-limit if continuous inference is required. Thus, the manual entries of instruction and output were implemented for GPT-4o to produce the output in this table.

4. Conclusion

The results showed the multimodal RAG can be used for retrieving and generating required data based on a PDF document that contains text, table and image, though verification by human is still required. The contribution of this study include:

- i. illustrating the use of multimodal rag for extraction and generation of specification related to the tasks of a bid writer or tender consultant;

- ii. benchmarking the performance of multimodal rag with basic rag and a widely used tool-GPT-4o for analysis; and
- iii. showcasing the use of internal models and software packages reduce data leakage to external parties.

The output accuracy can be enhanced using agentic RAG where the result for an initial response is evaluated by the same or other performant model. In addition, incorporation of various best practices in RAG such as reranking the retrieved chunks from vector store and using instruction-following models for inference can achieve better response from multimodal RAG. As this study do not have access to higher compute resources such as Graphical Processing Unit (GPU), utilization of larger models is limited. However, the objective of this study has been achieved by using the materials and methods as discussed. Future work may explore the methods to indicate the source of documents in the retrieved information for the instance of multiple PDF files as this study used a single PDF in the analysis. This is pertinent for companies that distribute and offer several products for their clients and partners.

Acknowledgments

The corresponding author is funded by the Public Services Department for his academic work.

References

- [1] Y. Sonoda *et al.*, "Diagnostic performances of GPT-4o, Claude 3 Opus, and Gemini 1.5 Pro in "Diagnosis Please" cases," *Jpn. J. Radiol.*, pp. 1-5, 2024.
- [2] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [3] E. F. Tsani and D. Suhartono, "Personality Identification from Social Media Using Ensemble BERT and RoBERTa," *Informatica*, vol. 47, no. 4, 2023.
- [4] L. Sun, J. Zhao, M. Han, and C. Xiong, "Fact-Aware Multimodal Retrieval Augmentation for Accurate Medical Radiology Report Generation," *arXiv preprint arXiv:2407.15268*, 2024.
- [5] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459-9474, 2020.
- [6] X. Wang *et al.*, "Searching for best practices in retrieval-augmented generation," *arXiv preprint arXiv:2407.01219*, 2024.
- [7] Y. Gao *et al.*, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, 2023.
- [8] Z. Tan, W. Wang, and C. Shan, "Vision transformers are active learners for image copy detection," *Neurocomputing*, vol. 587, p. 127687, 2024.
- [9] P. Joshi, A. Gupta, P. Kumar, and M. Sisodia, "Robust Multi Model RAG Pipeline For Documents Containing Text, Table & Images," in *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, 2024: IEEE, pp. 993-999.
- [10] Z. Cheng, J. Zhang, X. Xu, G. Trajcevski, T. Zhong, and F. Zhou, "Retrieval-Augmented Hypergraph for Multimodal Social Media Popularity Prediction," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 445-455.
- [11] M. Ajevski, K. Barker, A. Gilbert, L. Hardie, and F. Ryan, "ChatGPT and the future of legal education and practice," *The Law Teacher*, pp. 1-13, 2023.
- [12] G. Misuraca and C. van Noordt, "Overview of the use and impact of AI in public services in the EU," *Publications Office of the European Union: Luxembourg*, 2020.
- [13] S. Tonmoy *et al.*, "A comprehensive survey of hallucination mitigation techniques in large language models," *arXiv preprint arXiv:2401.01313*, 2024.
- [14] A. Bendigeri. "Ollama: A Lightweight, Extensible Framework for Building Language Models." <https://collabnix.com/why-ollama-is-damn-popular/> accessed October 8, 2024.
- [15] Meta. "Introducing Llama 3.1: Our most capable models to date." <https://ai.meta.com/blog/meta-llama-3-1/> accessed July 26, 2024.
- [16] Ollama. "llava-llama3." <https://ollama.com/library/llava-llama3> accessed October 8, 2024.
- [17] N. Team. "Nomic Embed Vision: Expanding The Nomic Latent Space." <https://www.nomic.ai/blog/posts/nomic-embed-vision> accessed October 8, 2024.
- [18] A. Gupta *et al.*, "RAG vs Fine-tuning: Pipelines, Tradeoffs, and a Case Study on Agriculture," *arXiv preprint arXiv:2401.08406*, 2024.