POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Multimodal RAG for Slide Presentations with Synthetic Data Generation and Anonymization

Supervisors

Prof. Daniele APILETTI

Dr. Simone MONACO

Candidate

Daniele MANSILLO

July 2025

Abstract

In recent years there has been a surge in the development and adoption of Retrieval-Augmented Generation (RAG) pipelines, as they constitute a cost-effective, flexible, and highly customizable way to leverage the advantages of LLMs on private and custom data. While modern RAG pipelines can work with almost any type of data, existing document processing systems focus predominantly on textual content, often ignoring visual elements. This text-centric approach may suffice when text constitutes the main information carrier, but it fails to extract all meaningful insights from documents like slide presentations, where content is equally distributed across text, charts, images, and tables that often interact with each other to convey complete information. Given the rising popularity and performance of multimodal models and the lack of substantial integration in RAG pipelines, we chose to bridge this gap by building an effective RAG pipeline capable of processing slide presentations in PDF format and accurately responding to queries requesting information available in different data modalities. The main goal of this research is to identify the optimal approach for embedding and retrieving multimodal slide content in order to provide high-quality answer generation capabilities, with a particular focus on questions that require information from multiple slides, while taking taking into account hardware constraints in order to explore and develop techniques that reduce computational cost without significantly compromising performance. To identify the optimal techniques at each stage of the pipeline we employed a multi-step approach, comparing, and if necessary developing, various techniques at each phase from the embedding through the answer generation. Due to the lack of readily available shared data, we designed two synthetic dataset generation techniques based on state-of-the-art multimodal LLMs. The first technique focuses on generating question-answer pairs from the content of multiple slides simultaneously, addressing a common limitation of existing methods that typically rely on single-image inputs. The second technique introduces a novel anonymization process that leverages recent multimodal LLMs to disguise sensitive or identifying information in slide presentations. This method is capable of anonymizing individual slides and eventually extending their context to generate coherent and complete synthetic presentations. This technique is capable of interpreting and reproducing slide content, including charts and visual layouts using LateX as a markup language, ultimately producing synthetic slides and presentations that are visually indistinguishable from authentic ones. The main contribution of the project is a robust RAG pipeline capable of embedding multimodal information extracted from slide presentations and generating accurate answers based on the extracted multimodal data. To support its implementation we introduce synthetic data generation and

anonymization techniques customized for slide-based documents. This research aims to support the advancing field of enterprise document intelligence by providing a comprehensive framework for multimodal content processing. The developed pipeline also offers practical solutions for organizations seeking to easily extract information from their slide presentations.

Table of Contents

List of Tables

List of Figures

Acronyms

RAG
: Retrieval Augmented Generation

NLP
: Natural Language Processing

LLM
: Large Language Model

VLM
: Vision-Language Model

COT
: Chain-of-Thought

Chapter 1

Introduction

The rise and rapid evolution of Large Language Models (LLMs) over the past few years was a major breakthrough in the field of artificial intelligence, especially for what regards Natural Language Processing (NLP) as a whole branch of research, from language understanding to generation. These models have demonstrated remarkable capabilities across a wide range of application fields, including question answering, text summarization and data extraction.

However, despite their impressive performance on general-purpose tasks, LLMs suffer from a significant limitation: they often hallucinate, generating false or misleading content, when queried about data not present in their training corpus. This limitation becomes critical in enterprise and private environments, where the data of interest is typically proprietary, constantly evolving, and not publicly available.

One way to mitigate this problem is to fine-tune the LLM on domain-specific data. While this can yield strong results, it presents several practical drawbacks: fine-tuning large models is resource-intensive in terms of time, computational cost, and data management. Moreover, the process must be repeated whenever the underlying data changes, making it infeasible for many real-world use cases involving frequently updated content.

This challenge has led to the development and widespread adoption of Retrieval-Augmented Generation (RAG) pipelines. These hybrid systems augment the capabilities of LLMs by integrating external information retrieval mechanisms. Rather than retraining the model, RAG pipelines retrieve relevant documents from a knowledge base and feed them to the LLM during inference, enabling it to generate informed responses grounded in the retrieved context. This approach is cost-effective, scalable, and well-suited for enterprise applications.

Initially, RAG pipelines were primarily designed for textual corpora. Over time, however, their scope has broadened to incorporate a wider variety of data formats, including tables, structured databases, and even images. Despite these

advancements, most RAG systems still operate in a predominantly text-centric manner, often neglecting the rich, complementary information encoded in non-textual elements such as diagrams, tables, and visual layouts. This shortcoming is particularly evident in slide presentations, which are widely used in professional and academic settings. In slides, meaning is often conveyed through a combination of text, visual elements, and spatial arrangements. A purely textual approach risks overlooking or misinterpreting this multimodal information.

With the recent rise in performance and accessibility of multimodal models, there is a growing opportunity, and necessity, to build RAG systems that can natively handle multimodal content. However, there is limited research and practical guidance on how to effectively integrate visual and textual features within a RAG framework, particularly in the context of slide-based documents.

This thesis aims to address this gap by designing, implementing, and evaluating a robust RAG pipeline capable of understanding and answering questions based on multimodal slide presentations in PDF format. The goal is to support accurate and contextually rich answer generation, especially for queries that require aggregating information spread across multiple slides and across different modalities (text, charts, tables, and images).

To tackle this challenge, we take a modular and comparative approach: we analyze multiple techniques for embedding, retrieval, and answer generation, carefully evaluating their performance and trade-offs. Where necessary, we propose novel techniques to enhance effectiveness or reduce computational cost without significantly sacrificing accuracy.

Given the scarcity of public multimodal QA datasets tailored to slides, we also introduce two custom synthetic data generation techniques. The first is designed to produce multi-slide question-answer pairs using state-of-the-art multimodal LLMs, while the second enables anonymization and realistic reconstruction of visual slide content, including diagrams and chart layouts, using LaTeX as a markup language. These tools allow us to simulate real-world scenarios while preserving data privacy and control over evaluation.

The main contributions of this work include:

- A complete, modular, and efficient multimodal RAG pipeline tailored to slide presentations;
- Novel synthetic data generation and anonymization techniques for multimodal slide documents, including the generation of question-answer pairs that require reasoning over multiple slides and the anonymization of individual slides using multimodal LLMs and extension of anonymized slides to produce realistic, coherent synthetic presentations;
- A comparative evaluation of answering techniques across both synthetic and

open-source datasets, covering the well-studied single-slide QA setting as well as the underexplored multi-slide QA scenario, including techniques such as aggregation, re-ranking via chain-of-thought prompting, and score normalization;

In the next chapter 2 we will first explore the key advancements that have enabled the development of high-precision embedding models and multimodal LLMs, which together form the foundation of modern RAG pipelines. In chapter 3 we will then detail the methodology adopted to design, implement and evaluate our custom pipeline from scratch. Finally, we present the anonymization techniques developed to transform real-world slide presentations into shareable synthetic datasets while preserving their structural and informational integrity.

Chapter 2

Related Work

The integration between Large Language Models and Retrieval-Based techniques led to the emergence of the Retrieval-Augumented Generation (RAG) paradigm, which has rapidly become a staple of Natural Language Processing (NLP) applications that require both scalability and precision. These systems combine the capabilities of LLMs to understand and elaborate texts with external and private knowledge sources leading to responses that are both user-accessible and grounded in factual context. At the same time, remarkable progress in the field LLMs have expanded their reasoning capabilties enabling them to operate over different data modalities spanning from pictures to multimodal data including text, charts, tables and images. Recent developments made it possible for LLMs to reason on multiple images at the same moment. Retrieval Techniques are the other pillar of RAG pipelines. Their performance has a direct impact on the effectiveness and on the quality of the generated answer. This field has seen a rapid advancement in the past years going from traditional methods such as BM25 to more advanced retrieval methods employing high-dimensional vector spaces capable of representing text, visual, and multimodal content with greater nuance. The advancement of these fields went hand-in-hand with the development of supporting datasets that allowed them to expand their scope. With more capable LLMs and retrieval techniques, more complex datasets were produced allowing researchers to test their Retrieval techniques on long texts, pictures, documents and their LLMs on complex tasks such as reasoning on multiple images.

In this Literature Review we begin by examining the evolution of Retrieval Techniques and LLMs. We then investigate how their progress led to more complex and sophisticated RAG pipelines and how the availability of diverse and task specific datasets supported this progress.

2.1 Evolution of Retrieval Techniques

Embedding and Retrieval Techniques are one of the two foundational components of the Retrieval-Augmented Generation (RAG) pipelines. Their performance in fact determines the relevance, precision, and effectiveness of the information employed by the LLM to generate the answers. While LLMs such as GPT [1], Gemini[2] and LLaMA [3] are capable of reasoning and text generation, the overall quality of the answer depends on how precise and relevant the retrieved context is. This section will focus on the developments of Retrieval Techniques from their employment in databases and electronic indexing systems to the most modern and advanced approaches. In the end, we will delve into the necessity for multimodal retrieval.

2.1.1 Background and Early Methods

Retrieval Techniques before the advent of AI-Powered Search Engines relied on keyword matching and sparse representations, employing different techniques to evaluate the relevance of this match. One of the first techniques to assess the relevance of keywords is inverse document frequency (IDF) [4]. This technique paired with term frequency (TF) produces a first measure called term frequency–inverse document frequency (TF-IDF) which offers a first measure of the importance of a word to a document in a collection or corpus. A high-level definition of tf-idf could be the following:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

where:

- $\text{TF}(t, d)$ is the term frequency of term t in document d.
- $\text{IDF}(t, D) = \log\left(\frac{N}{1+|\{d \in D : t \in d\}|}\right)$ is the inverse document frequency, with N being the total number of documents in the corpus D.

Bulding on TF-IDF, the Vector Space Model (VSM) [5] represents documents and queries in a high-dimensional space where each dimension corresponds to a distinct term in the vocabulary. Retrieval is then performed by calculating the similarity between the query and the document vectors. A typical similarity employed in the calculation is the cosine similarity:

$$\text{sim}(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{\|\vec{q}\| \cdot \|\vec{d}\|}$$

The Vector Space Model offers flexibility in how documents are encoded, supporting simple representation such as binary vectors, where each term in the vocabulary

is represented as 1-present or 0-absent in a document, and Bag-of-Words (BoW), where term frequency is used directly as the weight.

A more recent development of these retrieval models is the BM25 (Best Match 25) [6] which builds directly on the tf-idf model refining it by introducing mechanisms for term saturation and document length normalization. A high-level formula for the BM25 can be provided as follows:

$$\text{BM25}(q,d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{f(t,d) \cdot (k_1 + 1)}{f(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)}$$

$$\text{IDF}(t) = \log\left(\frac{N - n_t + 0.5}{n_t + 0.5} + 1\right)$$

Where:

- $f(t,d)$ is the term frequency of term t in document d,
- $|d|$ is the length of document d in words,
- avgdl is the average document length in the corpus,
- k_1 is a term frequency scaling parameter, typically set between 1.2 and 2.0,
- b is the document length normalization parameter, usually set to 0.75.

For their being simple, easily interpretable and computationally efficient, these systems are still heavily employed in search engines such as Apache Lucene, Apache Solr and ElasticSearch. However, their reliance on exact term matching and sparse representation of the corpus, limits their ability to represent and capture semantic relationships, and their performance in tasks requiring contextual knowledge, not to mention that these methods only function with textual data cutting out all other data modalities.

2.1.2 Dense and Neural Networks Enabled Embeddings

The limitations in capturing the context information have been addressed by innovative, neural networks based approaches in the past years. These techniques instead of representing the data as sparse vectors where each entry represents an exact word model the data as vector in high-dimensional spaces where the similarity between documents or between query and document can be measured using vector proximity

The first unsupervised techniques to employ this approach were **Word2Vec** [7] and **GloVe** [8]. These techniques were based on the distributional hypothesis which states that given a target word we could likely infer its semantic meaning by

its surrounding context. Word2Vec's key idea relies on training a neural network to predict the surrounding words of every word in a dictionary or on training a neural network to predict the target word given its surrounding words in the dictionary. The first training technique used by Word2Vec is called the Skip-gram model (Skip-Gram) which aims to predict the context given a word:

$$P(w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m}|w_c) = \prod_{j=-m, j\neq 0}^{m} P(w_{c+j}|w_c) \tag{2.1}$$

where w_c is the center word, m is the window size, and w_{c+j} represents the context words.

The second architecture is called Continuous Bag Of Words (CBOW) which aims to predict a word given its surrounding context.

$$P(w_c|w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m}) \tag{2.2}$$

The learned weight matrices obtained from these neural networks are then extracted and used as word embeddings. These embeddings are capable of capturing semantic relationships in the vector space.

GloVe (Global Vectors), as the name says, implements a more global approach by first constructing a word co-occurrence matrix that counts how often each pair of words appears together across the entire corpus. This matrix captures global statistical information about word relationships, rather than processing local windows one at a time like Word2Vec. GloVe then trains embeddings by optimizing the objective function:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}j - \log Xij)^2 \tag{2.3}$$

where X_{ij} represents the co-occurrence count of words i and j and $\tilde{w}_j$ are the word vectors, b_i and $\tilde{b}_j$ are bias terms, and $f(X_{ij})$ is a weighting function that prevents very frequent co-occurrences from being over-weighted.

These early approaches, even if capable of representing data in a more efficient way, suffered from the limitation of representing documents as simple aggregations of word vectors, failing to fully capture the meaning of the surrounding context.

The big leap forward came with the development of transformer-based models capable of generating contextual embeddings. **BERT** (Bidirectional Encoder Representations from Transformers) [9] led a revolution in the field by providing word representations that could have different embeddings depending on its surrounding context.

The architecture of BERT is built on part of the Transformer [10] architecture, specifically the encoder. The transformer encoder uses self-attention mechanisms

to capture the relationships between all words and positions in a sentence because they together convey the semantical meaning. The core of BERT's contextual understanding capabilities can be traced back to its multi-head self-attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.4}$$

where Q, K, and V are the query, key, and value matrices respectively, and d_k is the dimension of the key vectors. Multi-head attention allows the model to attend to different representation subspaces:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{2.5}$$

where each head is computed as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.6}$$

BERT introduces bidirectional training through its Masked Language Modeling (MLM) objective, where random tokens in the input are masked and the model must predict them based on both left and right context:

$$\mathcal{L}_{\text{MLM}} = -\sum i \in \text{masked} \log P(w_i|w_{\setminus i}) \tag{2.7}$$

where $w_{\setminus i}$ represents all tokens except the masked token w_i. Additionally, BERT employs a Next Sentence Prediction (NSP) task to understand sentence relationships:

$$\mathcal{L}_{\text{NSP}} = -\log P(\text{IsNext}|\text{CLS}) \tag{2.8}$$

where the [CLS] token's representation is used to predict whether two sentences follow each other in the original text. The final BERT training objective combines both tasks:

$$\mathcal{L}_{\text{BERT}} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{NSP}} \tag{2.9}$$

This pre-training allows BERT to learn different representations for each word based on the context. The generated representations are dependant on the context and on the corpus used for the training which means that BERT that can be fine-tuned for various downstream tasks employing different corpus, such as in our case, passage retrieval. BERT's impact on retrieval was very deep because it provided a way to encode entire passages taking into account the meaning of the context and not just the words themselves.

One of the first approaches to employ BERT as an embedder and retriever was DPR (Dense Passage Retrieval) [11]. In this case the model employs two different BERT encoders: one to encode the passages and another one to encore the user queries. The system aims to maximize the similarity between the query embeddings and the relevant passage embeddings while minimizing it for the irrelevant ones:

$$\text{sim}(q, p) = E_Q(q)^T E_P(p) \tag{2.10}$$

where E_Q and E_P are the query and passage encoders respectively. The training objective uses a contrastive loss function:

$$\mathcal{L} = -\log \frac{e^{\text{sim}(q,p^+)}}{e^{\text{sim}(q,p^+)} + \sum_{i=1}^{n} e^{\text{sim}(q,p_i^-)}} \tag{2.11}$$

where p^+ is the positive passage and p_i^- are negative passages for query q.

2.1.3 Advanced Neural Retrieval Architectures

Building upon the success of DPR, several advanced neural retrieval architectures have been developed to address specific limitations and improve performance. **ColBERT** (Contextualized Late Interaction over BERT) [12] introduced a novel approach that combines the efficiency of sparse retrieval with the effectiveness of dense representations. Instead of computing a single vector representation for each document, ColBERT generates token-level embeddings and performs late interaction through efficient vector operations:

$$\text{sim}(q, d) = \sum_{i \in q} \max_{j \in d} E_q^i \cdot E_d^j \tag{2.12}$$

where E_q^i and E_d^j represent the embeddings of the i-th query token and j-th document token respectively.

Another significant advancement is the development of learned sparse retrieval methods such as SPLADE (SParse Lexical AnD Expansion) [13], which combines the interpretability of sparse methods with the semantic understanding of dense models. SPLADE uses transformer models to predict importance weights for vocabulary terms, creating sparse representations that can capture semantic relationships through term expansion:

$$\text{SPLADE}(d) = \text{ReLU}(\text{MLP}(\text{BERT}(d))) \odot \log(1 + \text{ReLU}(\text{MLP}(\text{BERT}(d)))) \tag{2.13}$$

where $\odot$ denotes element-wise multiplication and the final representation is sparse due to the ReLU activation.

2.1.4 Multimodal Embeddings

Alongside the development of textual transformers and embedders, multimodal embedders quickly emerged to produce vector representations of data coming from different representations, enabling for similarity search across modalities. Among

the pioneering Vision-Language models **CLIP** (Contrastive Language-Image Pre-training) [14] demonstrated how it was feasible to represent joint embeddings for text and images in the same vector space. CLIP was originally trained on a large dataset of image-text pairs using contrastive learning, a technique still widely employed and influential.

$$\mathcal{L}\text{CLIP} = -\frac{1}{N}\sum i = 1^N \left[\log \frac{e^{\text{sim}(I_i,T_i)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(I_i,T_j)/\tau}} + \log \frac{e^{\text{sim}(I_i,T_i)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(I_j,T_i)/\tau}}\right] \tag{2.14}$$

where I_i and T_i are image and text embeddings, τ is a learnable temperature parameter, and N is the batch size.

CLIP paved the way for multimodal embedding models and extended and improved versions did not take long to arrive. Such as ALIGN [**Jia2021**] which made the training of such models more extensive and less human dependant by scaling the training data to over one billion of noisy image-text pairs.

Florence [15] afterwards produced a model that is capable of operating effectively on different types of tasks such as such as retrieval and VQA with minimal to no fine-tuning.

Recent developments proposed in BLIP (Bootstrapping Language-Image Pre-training) [16] improved the training even further balancing corpus size and captions precision. To retain only relevant text-image pairs for the training a holistic approach was adoped leveraging the image-captioning and image-caption quality evaluation capabilities of LLMs. BLIP-2 [17] afterwards improves the architecture by employing a frozen image encoder and a frozen large language model and allowing them to communicate efficiently via a lightweight Transformer module called Q-Former. The Q-Former takes a set of learnable query embeddings $\mathbf{q}$ and visual features $\mathbf{v}$ from the image encoder, producing output embeddings $\mathbf{h}$ such as:

$$\mathbf{h} = \text{Q-Former}(\mathbf{q}, \mathbf{v}) \tag{2.15}$$

2.1.5 Specialized Embedding Models for Documents

Documents, such as slides presentations, reports and forms present their unique challenges. This need led to the development of specialized embedding models. LayoutLM [18], with its following versions LayoutLMv2 [19] and LayoutLMv3 [20] develop and refine a framework to embed text, visual information and layout together. The overall embedding phase requires multiple steps where the document pages are parsed using OCR and the text as well as the images with their relative position in the page are all embedded together. In the paper these embeddings are referred to as 2-D position embeddings and image embeddings:

$$\mathbf{h}_i = \text{LayerNorm}(\mathbf{w}_i + \mathbf{p}_i^{1D} + \mathbf{p}_i^{2D}) \tag{2.16}$$

where $\mathbf{w}_i$ is the word embedding, $\mathbf{p}_i^{1D}$ is the 1D position embedding, and $\mathbf{p}_i^{2D}$ represents the 2D spatial position embedding for token i.

2.1.6 Latest Developments

The main challenge with the proposed multimodal embeddings is their lack of native encoding across modalities and the complexity of the processing techniques that employ intermediate steps such as OCR for text and images extraction

A recent model: Document Screenshot Embedding (DSE) [21] addresses these limitations by proposing a novel embedding methodology that unifies the different formats and modalities of documents into a single input format: screenshots. Unlike methods that require various tools to extract text and images, DSE directly encodes document screenshots using large vision-language models, this allows the embedding to natively preserve all the document including text, images, and their relative layout, without the need for any content extraction pre-preprocessing steps. DSE internally employs a bi-encoder architecture, where a document screenshot and a user text query are encoded into dense vectors in the same vectorial space using both vision and text encoders. The similarity between query and document can be afterwards computed as:

$$\text{sim}(q, d_{\text{screenshot}}) = \frac{E_{\text{text}}(q) \cdot E_{\text{vision}}(d_{\text{screenshot}})}{|E_{\text{text}}(q)| \cdot |E_{\text{vision}}(d_{\text{screenshot}})|} \tag{2.17}$$

where E_{text} and E_{vision} are the text and vision encoders respectively. A key enabler that allowed DSE to be so powerful and yet exceptionally small is the recent developoment of a new generation of lightweight large vision-language models (VLMs), such as Phi-3-vision [22], which employ novel techniques capable of representing images with more patches by cropping them into sub-images. This allows for the capture of more fine-grained textual and visual information compared to models like CLIP, which might only support fixed patch sequences.

Experimental results on the DSE paper show how the model outperforms traditional OCR-based text retrieval methods by more than 15 points in nDCG@10 in slide retrieval tasks. This specific result highlights how suitable this model is in scenarios where visual and textual elements are both equally relevant. Even for documents more focused on text such as academic papers, DSE still demonstrates satisfying performances, indicating its ability to effectively encode text directly from the image. We can see how DSE provides a model very much suitable to represent slide presentations while being careful about resource usage.

Another paper that adopts a similar approach with very promising results proposes the ColPali [23] embedder and retriever. The approach is similar to DSE where the final goal is to generate embeddings over the whole page as an image rather then separating its components with OCR and process them singularly.

As also mentioned in the paper, ColPali significantly outperforms traditional textual retrieval pipelines and contrastive VLMs on ViDoRe, a document based dataset proposed in the same paper, demonstrating the potential benefits of directly indexing visual features for efficient and effective document retrieval.

In addition to these screenshot-based approaches, our work also incorporates E5-V [24], a recently proposed universal embedding framework. E5-V leverages only text-pair data (so without using text-image pairs) to train a multimodal embedding model using MLLMs, relying on prompting strategies that enable it to jointly embed images and text into a shared space, without requiring explicit multimodal training data. This method enables significant reductions in training complexity (up to 95% less compute) while producing highly aligned text–image representations.

Another model we integrated in our pipeline is Stella [25], a distilled embedding model designed to match state-of-the-art performance at a fraction of the computational cost. Developed via multi-stage knowledge distillation from top-performing models on the MTEB [26] benchmark, Stella stands out for its high performance in text and multimodal retrieval while maintaining low latency and memory usage. This model is particularly useful in low-resource environments.

2.2 Evolution of LLMs

The development of Large Language Models (LLMs) represents one of the most crucial steps in artificial intelligence. Their introduction allowed Natural Language Processing to move from rigid statistical or rule-based models to models capable of nuanced reasoning and contextual understanding. In the latest years we have seen outstanding advancements in the field with models capable of operating and reasoning on multiple data modalities seamlessly and with exceptional performance. This retrieval was another key enhancer for the Retrieval-Augumented Generation task, as the improved capabilities of the LLM allowed the retrieved context to be translated to a meaningful answer in a more effective way.

2.2.1 Early Foundations and Transformer Architecture

The starting point for modern LLMs can be traced back to the introduction of the Transformer architecture [10] which led a revolution in the natural language processing through the introduction of the attention mechanism. Its architecture allowed the corpora to be processed in parallel and to handle long-range dependencies. Specifically, the self-attention mechanism allows the model to weigh the importance of different tokens within the sequence when encoding each one of them, resulting in representations that are contextually rich and more accurate.

Building on the Transformer architecture, GPT [27] introduced the autoregressive generation approach which became the foundation for modern large language models. This architecture, even if groundbreaking focused focused only on text comprehension and generation and operated with relatively limited parameter sizes and shorter context windows compared to current standards.

2.2.2 Scaling Laws and the Emergence of Large-Scale Models

During the following years model parameters and training data grew almost exponentially. This growth was driven by empirical observations of scaling laws [28], which demonstrated consistent performance improvements with increasing model size. GPT-3 [29], with its 175 billion parameters, showcased the impressive capabilities of LLMs that could now be used as general LLMs with complex reasoning capabilities, that were not tied to single domains anymore. The fine-tuning phase in most cases might have been replaced by a simple few-shot learning. This growth in the size of models continued over the years culminating in models such as PaLM [30] and even larger architectures that pushed the boundaries of what was computationally feasible.

2.2.3 The Multimodal Revolution

The transition from text-only to multimodal represents one of the most crucial steps for LLM develoment and for their integration with RAG pipelines.

The transition from text-only to multimodal capabilities represents perhaps the most significant recent advancement in LLM development. Early vision-language models like CLIP [14] demonstrated the feasibility of joint text-image understanding through contrastive learning, establishing foundational techniques for multimodal representation learning.

DocFormer [31] represents one of the first examples of multi-modal LLMs specifically fine-tuned for document understanding, specifically for the Visual Document Understanding task (VDU). This LLM is capable of processing text, vision and spatial features simultaneously. Among its key innovations, the model uses a novel attention mechanism that incorporates relative spatial relationships between document elements.

GPT-4V [1] marked a very significant moment in the advancement of multimodal AI, mainly due to its remarkable performance that enabled advanced reasoning in text and images seamlessly. This model demonstrated remarkable capabilities in visual question answering (VQA), document analysis, and complex multimodal reasoning tasks. The following release of GPT-4o further enhanced these capabilities with improved context handling and more efficient multimodal processing.

Google Gemini [2] adopted a different approach, natively training the models to be multimodal. They were in fact trained from scratch on multimodal data rather than adapting textual models. The latest series of Google Gemini models (2.5) is designed for multimodal tasks and is capable of processing text, images, audio and code in a unified multimodal manner.

Latest Anthropic's Claude models are also natively multimodal and very capable with mixed modalities.

2.2.4 Open-Source Multimodal Developments

On the Open-Source side many powerful models have been deveveloped and refined over the years. Starting with LLaVA [32] and LLaVA-NeXT [33]. These models were capable of working with images and texts in the same prompt in an effective way even if they were not natively designed to handle multiple images per prompt. These models were also relatively lightweight and executable even in resource constrained environments.

Alibaba recently presented a series of powerful open-source multimodal models: Qwen-VL [34]. These models are capable of reasoning across multiple input images and giving meaningful outputs even working with lower resolution source images. The recent Qwen2.5-VL [35] further improved the 2.0 version capabilities with better reasoning and understanding capabilities. The bigger versions of these models have achieved performance comparable with the leading commercial alternatives such as OpenAI GPT, Google Gemini and Anthropic Claude while maintaining open accessibility.

Another particularly relevant and recent open-source alternative is the InternVL series [36][37]. The largest version of the family, InternVL2_5-78B, is the first open-source MLLMs to achieve over 70% on the MMMU benchmark [38], matching the performance of leading closed-source commercial models. The recent InternVL3 [39] further enhances multimodal capabilities through native multimodal pre-training, by improving its ability to handle longer and more complex inputs, and optimizing performance at inference time, while also extending its practical capabilities to include tool use, graphical user interfaces use, and industrial applications.

Another model particularly relevant for resources constrained environments is Mini-InternVL [40] which was capable of achieving 90% of the performance of the full size model with only 5% of the parameters, making it an ideal model to deploy on consumer-grade hardware. This work highlighted the importance and most importantly the potential of model compression and efficient architectures for practical applications.

2.2.5 Current State and Future Directions

Only in the past two years we have seen the rise of many highly capable multimodal models and we see how native multimodal pre-training is becoming a norm in the development of new models. New models maintain their old textual abilities while expanding their reasoning and understanding on complex document modalities such as documents, audio, video and even 3D data [41]. We start seeing even LLMs capable of tool use interactions [42].

In our specific case, for RAG applications, modern multimodal LLMs offer unprecedented opportunities for developing powerful pipelines capable of processing and understanding complex documents and distillate information for the final user queries. The ability of new LLMs to simultaneously reason about textual content, visual elements, and their relationships enables more comprehensive information extraction and retrieval than was previously possible with text-only systems.

The parallel development of powerful commercial and open-source models ensures a competitive landscape capable of producing a wide variety of models with all kinds of new architectures and improvements. We see also how efficiency and hardware constraints are becoming relevant research topics which will drive the industry to produce new options for different deployment scenaraios.

These new open-source multimodal and efficient models are clearly the ideal components for processing complex documents like slide presentations, where information is distributed across multiple modalities and requires integrated understanding for a holistic approach.

2.3 Evolution of RAG pipelines

The development of Retrieval-Augmented Generation (RAG) systems radically changed the way in how language models access and use external knowledge. Throughout the years its architecture evolved and expanded from its foundational introduction as a text-only pipeline to the sophistical architectures of today capable of retrieval and generation across diverse data modalities.

2.3.1 Foundational RAG Architecture

The first RAG architecture and founding model was proposed in the paper "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" [43]. In this paper a hybrid approach that combines the static LLM capabilities with dynamic knowledge retrieved from external corpora. The original architecture was composed of two main components: a retrieval system based on DPR (Dense Passage Retrieval) and a sequence-to-sequence generator that generated the final answer based on the user query and on the retrieved context.

This original approach already targeted some key issues with LLMs such as always accessing up-to-date information and domain-specific knowledge not available natively in the chosen LLM. This was especially relevant on knowledge-intensive tasks or in tasks leveraging non-public data. This work laid the bases for modern RAG system: separation of knowledge storage and knowledge utilization, the importance of dense retrieval over sparse methods and the integration of retrieved context with the generation mechanism.

2.3.2 Early Extensions and Database Integration

After the first proposal of the RAG framework, researches recognized the potential and flexibility of the design. Throughout the years many extensions came that included different types of data sources. Among the first extensions some leveraged LLM capabilities also during the retrieval phase translating user natural language questions into SQL queries either through few-shot learning or through zero-shot learning, this enabled RAG systems to access structured informations stored in relational databases [40] [44] These approaches laid the groundwork for multi-modal data integration and demonstrated the flexibility of the RAG paradigm.

The subsequent development of more sophisticated retrieval mechanisms, including hybrid sparse-dense approaches and improved indexing end embedding strategies, further improved and fine-tuned the accuracy of the retrieval basedon the domain. These early improvements focused primarily on textual content, even if of different nature, but established architectural patterns that would later be foundational for multimodal extensions.

2.3.3 RAG on Tabular Data

Some specific tasks are worth of a mention due to their unique nature. One of these is the RAG task on tabular data which was first addressed using Dense Table Retrieval [45]. This research lays the foundation for open-domain Question Answering (QA) over tables. At its core it employs a two-step framework: a retriever that selects a small subset of candidate tables, followed by a machine reader that extracts the correct answer. Unlike approaches for free text, DTR modifies dense retrieval methods to better handle semi-structured tabular contexts. A key design choice was the inclusion of two distinct TAPAS Retriever [46] instances, one for the query and one for the table. Another key contribution of this paper is the production of a specific QA Dataset based on tabular data: Table QA dataset (NQ-TABLES), consisting of 11K examples where answers reside in tables.

A subsequent paper [47] though casted doubts about the effective necessity of using an ad-hoc architecture to process tabular data and showed how DPR performs well without any table-specific design or training, and even achieves

superior results compared to DTR when fine-tuned on properly linearized tables. The study explored three modules to explicitly encode table structures—auxiliary row/column embeddings, hard attention masks, and soft relation-based attention biases—but none yielded significant improvements. The conclusion is that table retrieval emphasizes content rather than table structure, suggesting that future developments can build upon successful text retrievers, and table-specific model designs should be carefully examined to avoid unnecessary complexity.

Among the most recent papers on the matter THoRR (TableHeader for Retrieval and Refinement) [48] seems to cast a new light on the topic by proposing a new methodology. At its core THoRR operates in two sequential phases: the retrieval phase, in which only the table headers are considered and retrieved and the refinement phase where the selected headers and information are filtered and used to create another table that will form the context to be fed to the LLM. This two phase method helps to reduce hallucinations in the final results.

2.3.4 RAG on Graphs

Another type of data that might be challenging to process with traditional methodology is represented by Graphs that given their intrinsic nature often provide information based on their structure and on the arrangement of nodes and relationships in addition of its content. When dealing with textual graphs these challenges cannot be natively addressed leveraging Graph Neural Networks (GNNs) or Graph Databases and that is the challenge that G-Retriever [49] aims to tackle. A key contribution is the introduction of a novel Graph Question Answering (GraphQA) benchmark, which targets complex reasoning across diverse applications, including common sense reasoning, scene understanding, and knowledge graph reasoning. Furthermore, it develops a methodology to directly tackle challenges like scalability and hallucination by selectively retrieving relevant subgraphs via a novel Prize-Collecting Steiner Tree optimization, thereby enabling complex question answering over real-world graphs. The framework integrates the strengths of Graph Neural Networks (GNNs), Large Language Models (LLMs), and RAG components.

2.3.5 RAG Improvement Techniques

Over the years many specific improvement techniques have been proposed to improve specific parts of the RAG pipelines. These techniques can have a different nature such as including new data in the context such as web data and relying on Knowledge Graphs [50] during the retrieval phase instead of relying only on dense embeddings. The mentioned paper also integrates a self-assessment mechanism for LLMs to evaluate the trustworthiness of generated answers, aiming to reduce hallucinations.

Another line of research instead focuses on a blend of searching techniques in the Retrieval phase [51] to extend the usual dense embeddings search with other strategies: keyword-based (BM25), dense vector-based (KNN), and semantic-based sparse encoders (Elastic Learned Sparse Encoder - ELSER) The main innovation of the paper is the proposal of hybrid queries (e.g., Cross Fields, Most Fields, Best Fields, Phrase Prefix) that leverage the strengths of each index.

Complementary approaches focus on optimizing the user query [52] using query expansion techniques. The main goal is to address the problem of hallucinations in RAG systems which are often induced by vague or ambiguous queries which fail to accurately capture user intent. The core idea is to refine queries using LLMs to achieve better precision in document retrieval.

Other researches [53] investigate the impacts of noise in the retrieval phase. A key finding of the research is that the retriever's highest-scoring documents that are not directly relevant to the query (distracting documents) negatively impact the LLM's effectiveness, leading to accuracy degradation, even with just one such document. The research hints the necessity to revisit common assumptions in Information Retrieval systems for RAG, inferring a trade-off between relevant and irrelevant documents. The optimal strategy appears to be retrieving a minimal set of relevant documents and then supplying random documents until the context limit is reached. The paper suggests that adding random documents might improve the LLM's precision by increasing attention entropy, potentially preventing "entropy collapse" and inducing enhanced accuracy.

2.3.6 Complete Multimodal Approaches

The necessity to develop multimodal RAG pipelines pushed researchers to find ingenious solutions and adopt different strategies for data retrieval and answer production.

MuRAR (Multimodal Retrieval and Answer Refinement) [54] is a good example of these innovative strategies. Its pipeline involves three main steps:

- First, text answer generation, where relevant text documents are retrieved and an LLM creates an initial text response.

- Second, source-based multimodal retrieval identifies and retrieves relevant multimodal data from the original documents, using contextual and LLM-generated text features for these elements.

- Finally, multimodal answer refinement uses an LLM to integrate the retrieved multimodal data into the initial text answer, producing a coherent and interactive multimodal output that goes beyond plain text responses.

MuRAR leverages the precision and efficiency of textual LLMs in a fully multimodal context, this strategy though presents a noted limitation that while source attribution ensures precision, it can result in low recall if relevant multimodal data is not in the same section or web document as the initially retrieved text snippet.

2.3.7 RAG Limitations

Traditional RAG systems though have their limitations and often find it challenging to handle complex real-world queries. They excel at retrieval punctual information but they may struggle with global knowledge. To overcome these limitations, GraphRAG [55] proposes a new approach that enables the user to query over large text corpora. GraphRAG first constructs a knowledge graph from source documents using an LLM, where nodes correspond to key entities in the corpus and edges represent relationships between those entities. It then partitions this graph into a hierarchy of communities of closely related entities, and an LLM generates community summaries in a bottom-up manner, recursively incorporating lower-level summaries. Finally, GraphRAG answers queries through a map-reduce process on these community summaries, generating partial answers that are then combined into a final global answer.

Along the same lines of the previously mentioned paper, the "Summary of a Haystack" (SummHay) task [56] a query-focused summarization benchmark designed to assess systems' ability to precisely summarize large sets of documents.

Another very common vulnerability in real-world documents is the presence of typos in the corpora which can lead to lower precision in the retrieval phase [57]. To consistently test their hypothesis the authors built a black-box adversarial attack method called GARAG (Genetic Attack on RAG) which employs different techniques to progressively introduce realistic typos in documents and see how much these typos can impact the performance of the RAG pipeline. The results show how GARAG consistently achieves high attack success rates (approximately 70%) across various QA datasets and significantly depletes the end-to-end performance of RAG systems, with an average reduction of 30% in Exact Match (EM). The findings highlight that lower perturbation rates pose a greater threat.

2.3.8 New Approaches for PDF files and documents

The Retrieval and Generation task on pdf documents and files with all its peculiarities is at the same time challenging a big opportunity for research given its immediate real-world impact. This field presents its own challenges such as data coming in different modalities and each modality can convey different information that needs to be also merged with the overall context.

To address the task of generating results from documents that contain relationships between images and texts MuRAG [58] is a solid proposal that that addresses this challenge by first extracting text from documents, and for images, it converts each page of the document into a single image. This step is designed to maintain the intrinsic relationship between images and text present on the same page. These "page-images" are then encoded with an attached detailed summary to provide rich, relevant context for LLMs. Embeddings of both text chunks and these processed "image-docs" are generated and stored in a vector store. At retrieval time textual and images can then be queried together.

VisRAG[59] also introduces a novel RAG paradigm specifically designed for multi-modality documents. Its core innovation is its VLM-based RAG pipeline, where the document is directly embedded using a VLM as an image, effectively bypassing the parsing stage and eliminating potential information loss or distortion. The VisRAG pipeline consists of a VLM-based retriever (VisRAG-Ret) and a VLM-based generator (VisRAG-Gen). VisRAG-Ret maps text queries and document images into the same embedding space and it can handle multiple retrieved pages using techniques like page concatenation or weighted selection for single-image VLMs, and direct multi-image input for capable VLMs. In the paper interesting and refined weighting techniques are employed to optimize the input that is passed to the VLM-based generator. The developed embedder proposed in VisRAG stored data more efficiently compared to the aforementioned ColPali embedder by employing a higher dimensional embedding space.

2.4 Datasets for Multimodal LLMs

The development of multimodal LLMs and multimodal RAG pipelines has been intrinsically linked to the evolution of training and evaluation datasets. Early datasets focused only on the link between text and image but rapid progress in the field, have created the necessity for more sophisticated benchmarks that require complex reasoning across multiple modalities.

2.4.1 Early Foundations and Simple Multimodal Tasks

The initial wave of multimodal datasets established fundamental benchmarks for image-text understanding. Datasets such as MS-COCO [60] and Flickr30K [61] provided basic image captioning tasks, where models were required to generate descriptive text for single images. Visual Question Answering (VQA) datasets like VQA v1.0 and v2.0 [62, 63] introduced question-answering capabilities for both images and videos but remained limited to straightforward visual recognition tasks. These early datasets typically required minimal reasoning and focused on object recognition, scene or video description, and basic attribute identification.

2.4.2 Transition to Complex Visual Reasoning

As multimodal models grew more powerful, the research community recognized the need for datasets that provided challenges beyond simple ones in a new way, forcing them to reason and. This transition marked the emergence of datasets that required deeper visual understanding and logical reasoning.

Datasets like PlotQA [64] and FigureQA [65] were among the first to propose new challenges such as complex visual reasoning over mixed data sources. These datasets in fact require the model to reason over charts and plots and extract information that requires multi-step reasoning and legend interpretation based on a mix of visual and textual data.

Further developments like TextVQA [66] and ST-VQA [67] proposed new challenges that required the model to read and understand text within an image and reason on the correlation between the two.

ChartQA [68] represents a significant milestone in this evolution, specifically targeting the comprehension and reasoning over charts. This dataset requires models to interpret various chart types and extract numerical information, perform calculations, and answer questions that demand understanding of relationships between data points. ChartQA requires the model to reason beyond simple visual recognition with tasks that involve mathematical reasoning and data interpretation, making it particularly relevant for slide presentations where charts often convey a big part of the message.

2.4.3 Documental Datasets and Layout Understanding

Documents in multimodal reasoning provided a unique challenge as textual data is very abundant and often easily parsable but images are still very much present and often convey key parts of the message. In more elaborated documents the layout and the relationship between text and images is often relevant and should be taken into account for effective reasoning.

DocVQA [69] was proposed as a benchmark for document scans visual question answering. This dataset was pretty challenging as it required models to understand document layouts, read printed text and handwritten text, extract information from tables, forms, and figures, and answer questions based on this multimodal information.

InfographicVQA [70] extended document understanding to infographics. In this case the challenge was slighly different as in infographics visual design elements and in general visual data convey almost the same amount of information as textual data. This dataset requires models to navigate complex layouts and integrate data from multiple visual and textual elements.

The DocVQA [71] dataset specifically addresses multi-page document understanding. In the paper the focus though is on single page and whole document

questions lacking examples referring to a specific subset of the pages. In any case the capabilities require to perform well on this dataset are particularly crucial for slide presentatio processing, where information is distributed across multiple slides and often the overall meaning of the presentation is crucial to obtain the right context.

2.4.4 Multi-Document Datasets and Cross-Modal Reasoning

With models becoming more powerful and natively capable of reasoning across different data modalities we see how benchmarks cross-modal reasoning are crucial in this development

Datasets like MultiModalQA [72] offer a unique challenge where in the dataset it is possible to find images containing various types of data including images, tables, and text passages. The information in this diversified dataset must then be processed to answer complex questions that cannot be resolved using any single modality or document.

The WebQA dataset [73] reiterates the challenge by presenting real-world web-based question answering scenarios where models must navigate multiple web pages, understand diverse content types and layout formatting and integrate information across all these different modalities to produce the final answer.

SEED-Bench [74] presents a comprehensive evaluation framework that spans multiple dimensions of multimodal understanding, including spatial reasoning, instance reasoning, visual reasoning, and text recognition. This approach provides a holistic assessment of model capabilities across diverse multimodal tasks.

2.4.5 Specialized Reasoning and Domain-Specific Challenges

With time the diversification and specialization of datasets gave birth to specific challenges pertaining to a single domain or focused on a very specific sub-part of multimodal reasoning.

MMBench [75] provides a comprehensive suite that helps to evaluate the single various aspects of multimodal understanding, including spatial reasoning, temporal understanding, and complex visual-linguistic comprehension.

ColPali [23], particularly relevant to document retrieval applications, introduces the Vidore dataset which contains only QA pairs on scientific pages papers. These pairs rely heavily on the combination of textual and visual data in the page. Similarly to Vidore also SPIQA (Scientific Paper Image Question Answering) [76], as the name says, proposes a dataset on scientifical papers multimodal reasoning.

2.4.6 Multi-Image Reasoning

The evolution toward more sophisticated multimodal reasoning has recently produced datasets that specifically target multi-image understanding. This represents a significant advancement from single-image analysis to scenarios where models must maintain coherent understanding across multiple visual inputs, a capability particularly crucial for document intelligence applications where information spans across multiple pages or slides. The evolution of these datasets was also empowered by the advancements in the multimodals LLMs that are now capable of processing multiple images simultaneously in an effective way.

A recent dataset that proposes a tough and innovative challenge is ReMI (Reasoning with Multiple Images) [77] which not only requires the model to understand and reason over the multimodal content presented in multiple images but forces the model to elaborate complex operations and comparisons between the two proposed images. These operations often involve a multi-step non trivial reasoning. The nature of the proposed images and problems is very diverse spanning from topics such as math, physics, logic, code, table/chart understanding, to spatial and temporal reasoning. The dataset contains 13 specific tasks including EmojiAlgebra, FuncRead, GeomShape, GeomCost, Collisions, Clocks, Schedule, Charts, CodeEdit, Isomorphism, Maps, RefCOCO, and IQ, providing a comprehensive evaluation framework for multi-image reasoning capabilities.

Another interesting yet very specific task is addressed by the ImageChain [78] dataset whcich proposes a unique multimodal challenge, the "next-scene description". The task consists of predicting the content of the next scene given a sequence of frames from a video.

These datasets really push the boundaries of what LLMs are capable of at the moment. These capabilities are central in the development of new LLMs capable of operating in increasingly complex tasks where the information is not only spread across multiple slides but it requires multi-step non trivial reasoning to get the right answer.

2.4.7 Challenges in Dataset Evaluation

The evolution of datasets towards more complex tasks where the answer is not trivial and short revelead the limitation of traditional evaluation metrics such as exact match, F1-score, BLEU [79] as ROUGE [80] as these metrics are not semantic but rely on n-gram overlaps. The creation of reliable evaluation metrics for complex tasks where semantic evaluation becomes more and more relevant is still an open research challenge.

2.4.8 Implications for Document Intelligence and RAG Systems

The evolution of the multi-modal datasets mentioned above has a direct impact on the development of sophisticated multi-modal Retrieval Systems, LLMs and as a result RAG pipelines. The evolution from simple datasets containing image-text associations to complex datasets where multi-image reasoning is the main task reflects the growing demands for datasets containing real-world applications where information is spread across different modalities and cannot be sourced only from one place.

Slides presentations represent a specific case where many different data modalities can contribute as the type of information in them is very varied, from text and images to charts and tables. Unlike documents, their primary purpose is not to store information but to present it to an audience, which often leads to each slide containing only a small amount of content. As a result, information about the same topic is often spread across multiple slides.

Chapter 3

Methodology

In this section we will cover the design choiches and their implementation in the developed multimodal RAG pipeline tailored for slide presentations. The system is built to support complex queries that require integrating information from both textual and visual elements and where also the layout plays a pivotal role. Given the absence of suitable open-source datasets for multi-slide multimodal QA, we constructed a custom dataset composed of business presentations and synthetic QA pairs to train and evaluate our system. Given the absence of suitable open-source datasets for multi-slides qa where retrieval and reasoning are equally relevant, we also propose a novel synthetic data generation approach to train and evaluate our system. The methodology focuses first on finding the best technique to embed the slides information for retrieval and then on seeking the best approach to generate the correct answer given the retrieved context, working in a resource-constrained environment.

3.1 Overview of the Proposed Pipeline

The final proposed RAG architecture consists of mainly 4 components:

- A presentation **preprocessing module**
- A **multimodal embedder** for each single slide
- A **retrieval mechanism** based on the embedder employed before to obtain the slides given a query in natural language
- An **optional pre-generation layer** employed to apply optimization techniques over the extracted slides
- A **generative model** to answer the query

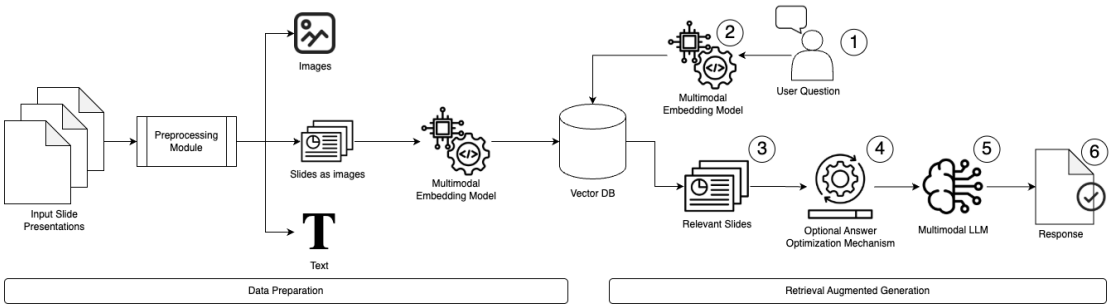


Figure 3.1: Visual Representation of the complete RAG pipeline.

A visual representation of the pipeline can be seen in Figure 3.1.

To select the best approach for each part of the pipeline we employed a modular approach by evaluating multiple candidate techniques at each major stage of the pipeline. For each stage we implemented and benchmarked several alternatives selecting the configuration that offered the best precision.

The final system then reflects a data-driven, performance-informed assembly of best-in-class techniques tailored to the multimodal slide domain. The following subsections detail each stage of the pipeline and the process that led us to the election of a specific component, from slide embedding to answer generation, along with the construction of synthetic datasets used for evaluation.

3.2 Synthetic QA Generation

To evaluate the pipeline, we used a custom dataset consisting of 1148 slides sourced from corporate presentations provided by a private company. These slides were rich in multimodal content such as bullet points, annotated charts, and interconnected tables. From this dataset, we generated 976 high-quality question-answer pairs by prompting a vision-language model (Claude 3.5 Sonnet) to create QA examples with explanations for individual slides. These were further validated using GPT-4o, which filtered out invalid or ambiguous questions. Each QA pair was linked to a specific slide, allowing us to assess retrieval precision by checking whether the pipeline returned the correct slide as the top result

```
System:
You are a question-answer pair generator.

User:
Your task is to write a factoid question and an answer given a
    context.
```

```
Your factoid question should be answerable with a specific,
   concise piece of information extracted directly from the
   context, preferably from charts and tables.
Your factoid question should be formulated in the same style as
   questions users.
This means that your factoid question MUST NOT mention something
   like "according to the given chart", "By the information in the
    text", etc.

Provide your answer as follows:

Output:::
Factoid question: (your factoid question)
Answer: (your answer to the factoid question)
Explanation: (explanation of the answer to the factoid question)

Now here is the context.

Context: Attached image
Output:::
```

Listing 3.1: Prompt used to generate the synthetic QA pairs on a single slide

3.3 Slide Preprocessing

To preprocess slides presentations efficiently we built a little Python library capable of reading either a folder full of presentations or a single presentation and from it extract text, images and pages as images and store this information in chromadb alongside with custom metadata. It also offered helpers to produce the embeddings and store them in chromadb.

3.4 Embedding Strategies for Multimodal Content

Slides pose unique challenges for retrieval pipelines due to their inherently multimodal structure. Unlike plain-text documents or even academic pdfs, slide presentations often contain interdependent elements such as titles, bullet points, images, tables, and charts, each contributing partial information. Moreover, the layout, visual hierarchy, and spatial relationships between these elements are often essential for interpreting the intended meaning.

To address this complexity, we adopted a comparative methodology: instead of committing to a single preprocessing and embedding strategy, we evaluated three distinct approaches, each reflecting a different level of multimodal integration

and computational cost. This allowed us to systematically assess how well each method preserved semantic relationships across modalities and supported accurate downstream retrieval and generation.

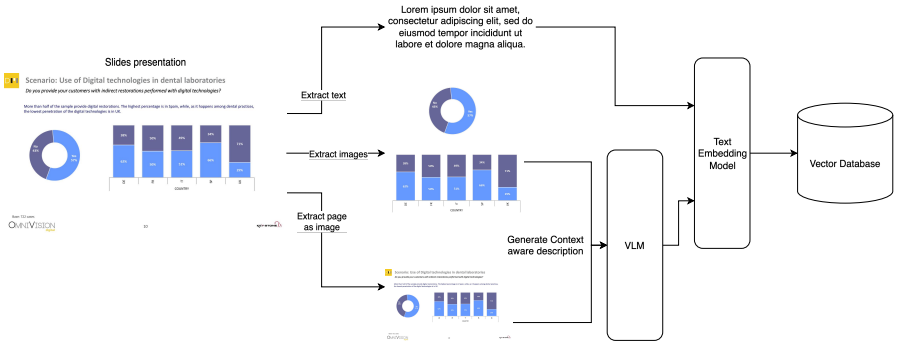


Figure 3.2: Text-Only Embeddings Strategy.

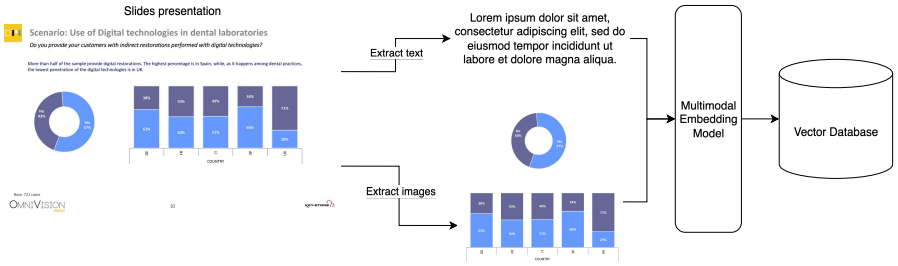


Figure 3.3: Separate Text and Image Embeddings Strategy.

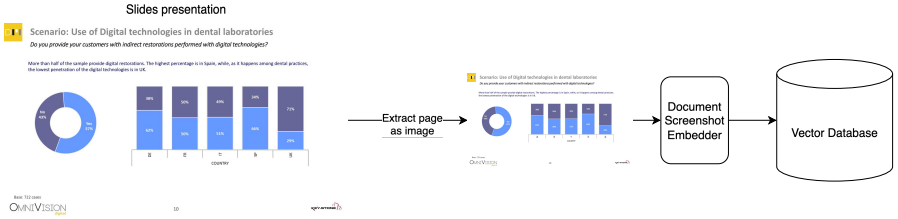


Figure 3.4: Whole Slide Visual Embedding Strategy.

The candidate strategies included:

- **Text-Only Embeddings** Figure 3.2: A simple extraction of textual content from each page followed by an extraction of all images and charts and an additional step where we asked an LLM to precisely describe the content

of the images. Then a text embedding using a pre-trained transformer was performed.

- **Separate Text and Image Embeddings** Figure 3.3: Leveraging a vision-language model individual elements (text boxes, tables, images) were parsed and embedded separately but using specific metadata in the vector database we were capable of extracting them both together.
- **Whole Slide Visual Embedding** Figure 3.4: The entire slide is embedded as a visual patch, capturing spatial and visual features alongside text.

3.4.1 Text-Only Embedding Technique

In the Text-Only Embedding Technique the process to embed documents follows multiple steps:

1. Extract text, images and whole pages as images from the pdf using pymupdf
2. Once all the images are obtained generate a description for each image given the page context. The prompt used to generate the descriptions is available at 3.2
3. Once all the image descriptions have been generated create the page text by concatenating the extracted text with the image descriptions
4. The last step is to embed page text, image descriptions and page text separately using the `NovaSearch/stella_en_1.5B_v5` embedding model

```
System:
You are a multimodal assistant tasked with generating precise and
   detailed descriptions of images.
You will describe images by leveraging contextual visual
   information provided from a related image.
Always focus on the details of the target image while ensuring the
    description is relevant to the context.

User:
The first image shows the entire page containing the second image.
Analyze the second image in the context of the first one.
Describe in detail the content of the second image.
Make your description specific and avoid generalizations.
Begin your description with phrases like 'The image contains...'
   or 'The image shows...'.
```

Listing 3.2: Prompt used to generate image descriptions given the image and the page

3.4.2 Separate Text and Image Embeddings Technique

The procedure for embedding text and images separately is much simpler compared to the text-only technique:

1. Extract text and images from the pdf using pymupdf
2. Embed page text, and images with the page text for context separately using the `royokong/e5-v` embedding model

3.4.3 Whole Slide Visual Embedding Technique

This technique is also quite simple and involves a few steps:

1. Extract text and whole pages as images from the pdf using pymupdf
2. Embed page text, and page images separately using the `MrLight/dse-qwen2-2b-mrl-v1` embedding model

Also in this case we decided to keep the textual embeddings to see how they would have influenced the retrieval process. Later in the processing they were not used anymore as their usage did not improve the quality of the retrieval.

3.4.4 Each Technique's Pros and Cons

Each technique presented different challenges and trade-offs:

- **Text-Only Embeddings**: This method enables fast retrieval and is compatible with lightweight, well-established language models. Moreover, since all the information is stored as text it could enable the architecture to employ text-only models also for the generation phase; this would make the pipeline capable of running on resource constrained environments. The trade-off comes with the embedding phase which requires powerful machines and generating the images description is a vary lengthy operation which may still miss some specific information useful for particular questions.
- **Separate Text and Image Embeddings**: In this setup, text and images are processed independently. It benefits from specialized models for each modality and allows higher-resolution image embeddings since text, layout and other boilerplate data isn't embedded as part of the image itself. However, in this method we lose track of the original visual layout of the slide which can be crucial for interpreting content correctly.

- **Whole Slide Visual Embedding**: This last approach is the most naive and the simplest. It leverages the latest developments in the VLM world with powerful embedding models. It allows us to keep all the information in the image but on the other side the text is stored as part of the image and not as a string and the overall image can contain a lot of unrelevant data which still takes space and pixels that make the computations more cumbersome. The primary strength of this approach lies in retention of all the information related to the layout and spatial structure of the slide content.

The evaluation for the retrieval mechanism was very straightforward: for each query the top 1 retrieved page was compared against the ground truth, defined as the original slide used to generate the corresponding QA pair. The technique that achieved the highest top-1 precision was elected as the preferred technique. The expected result for each query was the slide originally used to generate its QA pair.

The overall dataset was composed of 976 pages and in the end we obtained 976 questions.

Among the tested approaches, the "Whole Slide Visual Embedding" strategy achieved the highest Top-1 precision and was ultimately selected for the final pipeline. A summary of these results is shown in Table 4.1.

3.5 Answer Generation Module

After assessing the best embedding and retrieval technique, we focused on identifying the best technique to generate answers given the extracted context. Given the established retrieval setup, we explored and compared several optimization strategies aimed at producing more coherent answers, especially in resource-constrained environments. As a baseline, we used the direct answer generated from the top-1 retrieved slide without further processing.

All optimization techniques were evaluated on the generated synthetic dataset containing QA pairs that referred to a single slide only, as this subset provided a sufficiently large and consistent set of examples for reliable experimentation.

```
System: You are an advanced answer generation assistant in a
    Retrieval-Augmented Generation (RAG) pipeline.
Your task is to provide accurate, concise, and contextually
    relevant answers to user queries.
You will receive retrieved images and a specific question to
    answer.
Use only the provided context to formulate your response, and do
    not make assumptions beyond the given information.
If the context is insufficient to answer the query, clearly state
    that more information is needed.
User:
```

```
<Query>
<Context images>
```

Listing 3.3: Prompt used to generate the single answers in the baseline

3.5.1 Answer Generation Optimizations

The key differences among the evaluated techniques lie in how the retrieved context is pre-processed before being passed to the answer generation module. Some of these techniques employ ad-hoc selection mechanisms to elect a single, most relevant image from the top retrieved set, aiming to replace the top-1 baseline with a better suiting slide. Other techniques instead generate multiple answers from several candidate slides and then apply an aggregation strategy to produce the final answer.

3.5.2 Each technique in detail

The following sections provide a detailed breakdown of each answering technique, outlining the prompts used for response generation and the aggregation mechanisms employed to construct the final answer.

Technique 1: Slide Image + Text Context

In this technique for each of the top-n retrieved pages, only the image from one page is used while the accompanying text comes from all the n pages. This process is repeated for each of the n retrieved pages, generating n candidate answers, each leveraging a different visual context but enriched by the same textual surroundings. In the final step, all intermediate generated answers are aggregated into one single final answer. Below you can see the prompt to generate the single intermediate answers 3.4 and to generate the final aggregated answer 3.5.

```
System: <Same prompt as in the baseline>
User:
This is the textual content of the top_k retrieved pages, if
    useful use it to generate the answer:
top_1 page text: Lorem ipsum dolor sit amet, consectetur
    adipiscing elit, sed do eiusmod tempor incidunt ut labore et
    dolore magna aliqua
top_2 page text: ...
...
top_k page text: ...

Here the user query: <user query>
```

Listing 3.4: Prompt used to generate the single answers in technique 1

```
System: You are an advanced AI tasked with synthesizing a final response from multiple retrieved and generated responses.
These responses were produced using a Retrieval-Augmented Generation (RAG) pipeline, where the first responses are based on images that matched more strongly with the user's initial query.

**Instructions:**
- Analyze the **<n> responses** provided below.
- **Prioritize information from the first responses**, as they are derived from more relevant image matches.
- Resolve contradictions by emphasizing the most reliable or widely agreed-upon details.
- Remove redundancy while ensuring no key information is lost.
- Maintain **clarity, coherence, and factual accuracy** in the final response.
- If there are **uncertainties or conflicting claims**, highlight the most probable

User:
Responses from RAG (First responses are image-prioritized):
<response 1>
<response 2>
...
<response n>

Here the user query: <user query>

Generate a well-structured, synthesized response that integrates the best elements of the retrieved responses, giving priority to early responses based on image relevance while incorporating useful details from later responses.
```

Listing 3.5: Prompt used to generate the aggregated answer in technique 1

Technique 2: Progressive Accumulation

All top-k retrieved slides are resized to fit input constraints, and progressively grouped to generate k answers. Specifically, the i-th answer is generated using the top-i pages ($1 \leq i \leq k$), so that the model sees increasing context in each step. The reason is that higher-ranked slides are likely more relevant, and their influence is weighted more heavily by appearing in more generated answers. Final answer aggregation is then performed across these k outputs. k is a hyperparameter but in this case we used a k=5.

The prompt used to generate the intermediate answers is the same one used to generate the answer for the top-1 baseline: Listing 3.3 while the prompt used to

generate the final aggregated answer is the same as in technique 1: Listing 3.5.

Technique 3: Confidence-Based Selection via VLM-Evaluation (raw and Chain Of Thought variants)

In this technique each of the top-n retrieved slides is processed independently. For each one, the model is asked to assess its own confidence in whether that single image can comprehensively answer the question. It does so by generating a JSON object containing a confidence score and an explanation (reasoning). Empirical tests showed that requiring an explanation significantly improves the consistency of the confidence scores. The image with the highest confidence score is selected, and its corresponding answer becomes the final output. In the event of a tie, the highest-ranked image is chosen. To show the difference between the version with and without reasoning we tested both.

Below the prompt used to obtain the confidence with 3.7 and without 3.6 leveraging the Chain Of Thought:

```
System: <Same prompt as in the baseline>
User:
Please evaluate the confidence (0-100) that the following question
    can be answered using the provided image in the context:
Query: <query>
Answer only with a number.
```

Listing 3.6: Prompt used to generate the confidence metric in technique 3

```
System: <Same prompt as in the baseline>
User:
Think step by step before answering.
1. Identify the key elements needed to answer the query. What kind
    of information does the query require?
2. Examine the given image and determine whether it contains the
   necessary details.
3. Compare the query requirements with the image content. Does the
    image fully, partially, or not at all provide the needed
   information?
4. Based on this reasoning, assign a confidence score from 0% to
   100%.
Now, as your answer provide your reasoning followed by a
   confidence score as a single number, in the following json
   format: {"reasoning": <reasoning>, "confidence": <confidence>}
Query: <query>
```

Listing 3.7: Prompt used to generate the confidence metric in technique 3 leveraging the Chain Of Though

Technique 4: Distance + Perplexity-Based Scoring (Raw and Normalized Variants)

In both of the last two techniques, the model generates an answer for each of the top-n retrieved pages individually. For each answer, two scores are extracted:

- The retrieval distance from the retriever.
- The LLM perplexity score, measuring how confidently the model produced that answer. To calculate the perplexity score a relatively small textual model has been employed: `Qwen/Qwen2.5-1.5B-Instruct`

In the first variant, the raw (unnormalized) product of retrieval distance and perplexity is used to select the answer with the lowest combined score. In the second variant, both scores are normalized before multiplication to account for scale differences. The answer with the lowest normalized combined score is selected as final. *(This technique is inspired by the VisRAG approach.)*

The prompt used to produce the candidate answers is exactly the same as the one used in the baseline 3.3.

Compound Metric Calculation Given the retrieval distances d_i and the model perplexities p_i for each of the top-n retrieved slides, we compute two compound metrics: one unnormalized and one normalized.

To compute the normalized compound metric, we first normalize both the distances and the perplexities across the top-n candidates. The normalized distance $\hat{d}_i$ and perplexity $\hat{p}_i$ are defined as follows: $\hat{d}_i = \frac{d_i - \min_j d_j}{\max_j d_j - \min_j d_j}$ and $\hat{p}_i = \frac{p_i - \min_j p_j}{\max_j p_j - \min_j p_j}$.

After normalization, we compute the compound scores by taking the elementwise product of distances and perplexities. For the **unnormalized** case: $s_i = d_i \odot p_i$ and for the **normalized** case: $\hat{s}_i = \hat{d}_i \odot \hat{p}_i$.

To select the best candidate, we find the index corresponding to the minimum compound score in each case: $i^*_{\text{unnorm}} = \arg\min_i s_i$ and $i^*_{\text{norm}} = \arg\min_i \hat{s}_i$

The final selected answers are determined by the indices that minimize the respective compound scores. Specifically:

- The answer selected using the unnormalized metric is given by $\text{Answer}_{i^*_{\text{unnorm}}}$.
- The answer selected using the normalized metric is given by $\text{Answer}_{i^*_{\text{norm}}}$.

3.5.3 Answers Evaluation

To evaluate the answers produced by the top1 baseline and by the optimization techniques we first attempted using some traditional techniques leveraging the RAGAS [81] Python framework. To filter out the techniques that were clearly not suitable we manually selected two examples:

- One pair where the answer produced by the RAG pipeline using the top-1 retrieved slide was factually correct, closely aligned with the original dataset-generated answer (produced by Claude 3.5 Opus).
- One pair where the top-1 RAG answer was factually incorrect.

First we tested traditional text-based techniques:

- String Presence
- BLEU Score
- ROUGE Score
- Exact Match

However, they all produced sub-optimal results often not recognizing a correct answer when there were variations in wording, sentence structure, or phrasing, despite the semantic meaning being preserved.

Afterwards, we evaluated different metrics available in the RAGAS framework that offered LLM supported metrics:

- LLMContextRecall
- FactualCorrectness
- SemanticSimilarity
- Faithfulness

The LLM model we used to evaluate the correctness was GPT-4o But also in this case, when the RAG-generated answer conveyed the same meaning as the reference, it was occasionally marked as incorrect, indicating limitations in these LLM-based metrics.

After observing the limitations of both traditional and RAGAS LLM-based evaluation methods, we adopted an alternative approach using LangChain's LLM-based QA evaluator, which yielded more reliable results aligned with human judgment. Specifically, we used the `qa` evaluator module provided by LangChain [82], backed by GPT-4o.

3.5.4 Impacts on Latency

It is clear to see how the optimization techniques have a deep impact on the latency of the answers. Each technique has to produce a variable number of answers before being able to output the final answer. Each technique is then quite slower compared

to the simple top1 answer. A summary table containg a comparison of the passes requred for each methodology is available in Table 3.1.

The adoption of optimization techniques in the answer generation pipeline introduces significant variations in response latency. Unlike the baseline approach, which generates a single answer based on the top-1 retrieved slide, the proposed techniques often involve generating multiple candidate answers or performing additional reasoning steps before obtaining the a final response.

Each method differs in the number of required model invocations:

- Some techniques (e.g., confidence-based selection or perplexity-weighted aggregation) generate answers independently for each of the top-n retrieved slides and then apply a selection or aggregation mechanism.

- Others require sequential evaluations, such as estimating confidence scores for each possible input.

- Techniques that aggregate intermediate answers using LLM-based aggregation introduce additional passages through the model, adding even more latency.

As a consequence, the latency overhead of these techniques can be substantial when compared to the baseline.

Before adopting them in a real-life scenario it is critical to determine the trade-off between precision increment and latency.

3.5.5 Best Techniques Selection

Following the evaluation of all proposed optimization strategies, we selected a subset of techniques that consistently demonstrated the highest answer accuracy.

The techniques that showed promising results were:

- **Technique 1: Slide Image + Text Context**

- **Technique 2: Progressive Accumulation**

- **Technique 3: Confidence-Based Selection via VLM-Evaluation with Chain Of Thought**

- **Technique 4: Distance + Perplexity-Based Normalized Scoring**

Technique 3 without Chain Of Thought and Technique 4 with the un-normalized scoring yielded sub-optimal results and were not kept for the other tests

Table 3.1: LLM Passes per Answer Generation Technique

Technique	LLM Passes	Description of Passes
Baseline (Top-1 Only)	1	Single pass using the top-1 retrieved slide to generate the answer.
Technique 1: Slide Image + Text Context	$n + 1$	One LLM pass per top-n page, followed by one aggregation pass over all answers.
Technique 2: Progressive Accumulation	$k + 1$	One LLM pass per subset using top-1 to top-i slides ($i = 1..k$), plus one aggregation step.
Technique 3: Confidence-Based Selection via VLM-Evaluation	$n + 1$	One LLM pass per image to generate confidence and reasoning, followed by one final answer generation using the best candidate. Here we are not counting the passes to calculate the perplexity
Technique 4: Distance + Perplexity-Based Scoring	n	One LLM pass per top-n image to generate the answer, one additional computation step (not LLM) to rank and select.

3.6 Multi-slide Questions Generation

After building a solid RAG pipeline for single slide answering we decided to focus on building a really innovative dataset focusing on questions that were possible to answer only with information coming from multiple slides simultaneously.

This part involved many preparatory steps as it was not possible to simply pick two random slides in the corpus and ask a multimodal LLM to generate a QA pair based on both slides as the content of the slides would have probably been totally disconnected.

To overcome this problem we employed a series of refinement techniques to obtain only meaningful couples.

3.6.1 Automatic Section Extraction

The first step to make the multi-slide questions generation more efficient consisted of automatically dividing the document in different sections. As the structure of the documents was somehow similar since produced by the same company and they were reports focusing on different years and geographical areas. The first step to do so was to parse all the text contained in the document using pymupdf (in

this phase we exclude the images since the section scope can be inferred by the text only and working with the images would have made the process incredibly slow and resource intensive.

Then we passed the whole text of a slides presentation to the LLM (in this case Claude Sonnet 4) with the following prompt 3.8:

```
User:
You are an expert document analyzer.

Please divide the document below into logical sections.

For each section, return:
- Section title: if the title is already available in the document, DO NOT change it
- Start page index
- End page index
- A short summary of the section

Output must be a JSON array like:
[
{
    "section_title": "...",
    "start_page_index": 0,
    "end_page_index": 3,
    "summary": "..."
},
...
]

Document content:
Page 1: Lorem ipsum dolor sit amet
Page 2: ...
...
Page n: ...
```

Listing 3.8: Prompt used to extract the sections from the documents

What was obtained from this step was a json for each document with the following structure 3.9:

```
{
    "section_title": "1. Statistical note",
    "start_page_index": 2,
    "end_page_index": 3,
    "summary": "Contains methodology details, glossary of
   statistical terms (confidence level, confidence interval), and
   sample distribution across European countries (France, Germany,
    Italy, Spain, UK) with 1,018 total interviews conducted March-
   April 2023."
}
```

Listing 3.9: Sectioning result example

For each document we also manually appointed some additional metadata: report year, report country/geographical area.

3.6.2 Section Pairing

The initial step in generating QA pairs from two distinct input slides involved identifying semantically similar sections across different documents. Once the JSON structure containing all document sections was prepared, we proceeded to pair sections based on semantic similarity. The following pre-processing steps were applied to each section:

1. The section title and summary were concatenated to form a single textual representation.
2. This concatenated string was encoded using a Sentence Transformer to obtain dense vector embeddings.
3. We computed pairwise similarity scores between all section embeddings from different documents.
4. Section pairs with a cosine similarity score above 0.9 were retained. This threshold was chosen empirically, as it yielded a satisfactory number of high-quality pairs. In future iterations, this threshold can be exposed as a tunable hyperparameter to further optimize results.

3.6.3 Pages Pairing

After identifying semantically similar sections, the next step involved pairing individual pages from those sections. The goal was to find pages that convey related but not identical content, ensuring contextual alignment while avoiding redundancy.

The following technique was used:

1. For each section pair, we retrieved all page embeddings belonging to the two sections.
2. We performed a Cartesian product comparison, evaluating the cosine similarity between every possible pair of pages, one from each section.
3. We retained only the page pairs whose similarity score fell between 0.7 and 0.9. The upper bound of 0.9 was intentionally set to filter out pages that are likely near-duplicates or exact matches.

This method allowed us to identify page pairs that are contextually aligned yet distinct, increasing the likelihood that their combined content would support meaningful question-answer generation without repetition.

3.6.4 Pair Questions Generation

Once page pairs were identified, the next step involved generating question-answer (QA) pairs based on these slide combinations. Given the increased relevance of the contenxt in this task, we also included relevant metadata specifically, the report year and country which are critical in market research and highly informative for end users.

We used Claude 4 Sonnet as the language model for this generation task. To ensure high-quality QA pairs grounded in both slides, we employed the following structured prompt 3.10:

```
User:Your task is to write a factoid question and an answer given a context.
You are provided with two slide images from market research reports with some metadata about the reports.
Your factoid question should be answerable with a specific, concise piece of information extracted directly from the context, preferably from charts and tables.
The question must relies equally on information from both slides.
Your factoid question should be formulated in the same style as questions users.
This means that your factoid question MUST NOT mention something like "according to the given chart", "By the information in the text", etc.
In the question formulation you can use the provided metadata.

Provide your answer in json format as follows:

Output:::
{Factoid question: (your factoid question)
Answer: (your answer to the factoid question)
Explanation: (explanation of the answer to the factoid question)
}

Now here is the context.

Context: Attached images
**Slide 1**
Document name: <slide_1 document name>
Country: <slide_1 document country>
Year: <slide_1 document year>

**Slide 2**
```

```
Document name: <slide_2 document name>
Country: <slide_2 document country>
Year: <slide_2 document year>

Output:::
```

Listing 3.10: Prompt employed for Slides Pairs Questions Generation

The inclusion of an explanation in the output leverages the model's chain-of-thought (COT) capabilities. This not only improves the factual grounding of the generated QA pairs but also serves as a valuable debugging and validation tool during dataset inspection.

3.7 Multi-slide Questions Answering

To evaluate multi-slide question answering, we adopted a simple baseline using the top-2 retrieved slides to serve as a naive reference point. We then adapted a subset of the optimization techniques previously introduced, tailoring them specifically for scenarios in which relevant information may be spread across multiple pages.

These selected techniques were chosen for their capacity to leverage broader context while maintaining reasonable complexity. In the following subsection, we describe each of them in detail, highlighting the specific adaptations made for the multi-slide case.

3.7.1 Each Technique in Detail

In this section, we describe the 3 optimization techniques adapted for the multi-slide question answering scenario. Each technique is designed to incorporate multiple slides in the context when possible.

Technique 2: Progressive Accumulation

This technique did not change from the description provided in section 3.5.2.

Technique 3: Confidence-Based Selection via VLM-Evaluation with Chain Of Thought

The evaluation phase in this technique happens in the same exact way as explained in section 3.5.2. To adapt it to the multi-slide context, we introduce a tunable hyperparameter called `confidence_range`, which allows the inclusion of all slides whose confidence scores fall within the interval `[highest_confidence - confidence_range, highest_confidence]`.

This enables the model to consider multiple high-confidence slides—not just the single top one while still prioritizing those deemed most relevant. If more than one slide is selected, they are resized as necessary and jointly passed to the VLM for the final answer generation.

Given the negligible computational overhead, we retain both the normalized and unnormalized versions of this technique for this comparison.

Technique 4: Distance + Perplexity-Based Scoring

Also in this case the initial steps are exactly the same as explained in section 3.5.2. To allow for selection flexibility, we introduce a tunable hyperparameter `confidence_range` (expressed as a percentage). All candidates whose M_i falls within

$$[\min(M),\ \min(M) + \texttt{confidence\_range}\% \cdot \min(M)]$$

are retained.

If more than one candidate satisfies this criterion, their corresponding answers are aggregated using the same aggregation prompt used in 3.5.2.

3.8 Presentations anonymization

The last step we followed was the presentations anonymization. Since we were using private data not suitable for sharing and given that available data online for this type of operations (multimodal slides presentation answering on multiple pages) was not available we decided to create a technique that allowed the creation of anonymized data starting from multiple slides together.

This approach was followed because we found that anonymizing a single page often led to a harsh domain shift or the context tended to change too much. Changing the pair allowed us to have a good starting point to generate new qa pairs and allowed us to expand each one of them to generate a presentation. This approach also inserted more variance in the generation as the VLM perceived the couples as different from the single pages giving us more variability in the context shift.

To perform the anonymization a small pipeline with the following steps was built:

1. Given the slide couples as images and their metadata

2. Feed an LLM (in this case gemini-2.5-pro as it combined cheap prices with native multimodal capabilties. We also tested smaller models such as gemini 2.5 flash but the results were suboptimal) with the couple and a custom prompt to generate two slides in LateX beamer

3. After the generation some python postprocessing was applied to ensure that the generated images would fit into a page

We performed this anonymization operation with all our slide couples.

Afterwards the obtained LateX code was compiled in pdfs and each single pdf page was then parsed as images and evaluated by an external VLM (Gemini 2.5 pro in this case) which checked if all the data was visible, if any sections overlapped with each other and if the layout was consistent.

The final step in our process involved the anonymization of slide presentations. Since the employed dataset contained proprietary or confidential information unsuitable for public release and considering that no publicly available datasets support multi-slide multimodal retrieval and answering tasks we developed a custom anonymization method to produce, given real-world presentations, realistic but shareable content.

In our approach we focus on anonymizing slide pairs instead of single slides as we found that altering only a single slide often results in a strong domain shift or in a complete change of the content. On the contrary, transforming slide pairs enabled the resulting slides to better retain structural and topical relationships, offering a more reliable foundation to eventually generating high-quality QA pairs. Moreover, treating slide pairs together resulted in a greater variability, as the VLM perceived the couple as new even if the single components were already anonymized in different combinations.

3.8.1 Anonymization Pipeline

To anonymize the slides, we built a lightweight pipeline composed of the following steps:

1. **Input**: A pair of slide images along with their associated metadata (e.g., document name, country, year).

2. **Generation**: The pair was passed to a VLM specifically `Gemini 2.5 Pro`, selected for its native multimodal support and low inference cost. A custom prompt instructed the model to generate two LaTeX Beamer slides representing anonymized data with a similar structure but with changed domain and entities.

3. **Post-processing**: After generation, a Python post-processing script ensured layout consistency and adjusted formatting to ensure that the rendered slides would fit cleanly within standard page boundaries. This process is not error proof and can for sure be enhanced in future.

The prompt employed for the slides anonymization was the following:

```
User:
You are an expert LaTeX Beamer presentation designer and anonymization specialist.

Your task is to analyze the two provided slide images and generate a single, complete LaTeX Beamer document. This document MUST contain exactly two slides (`\frame` environments), where each slide is an anonymized version of one of the input images.

Follow these rules for anonymization:
- Replace all real-world country names with fictional, plausible ones (e.g., "Country A", "Region B").
- Modify all numbers, percentages, and statistics plausibly and randomly, ensuring they still make sense in context (e.g., if a chart shows growth, the new numbers should still show growth).
- Substitute all specific labels, headings, company names, product names, and organization names with fictional but coherent alternatives.
- Remove or replace any other direct real-world references.

For each of the two slides in the generated Beamer document:
- It must be a complete `\frame` environment.
- It must accurately reflect the key visual and textual information present in its corresponding original image.
- Use `\frametitle{}` for a clear, anonymized title.
- Include appropriate LaTeX elements such as bullet points (`\itemize`), numbered lists (`\enumerate`), blocks (`\begin{block}`), tables (`\begin{tabular}`), or simple text, to best represent the content.
- Ensure the content fits professionally within a single page, adapting the original layout as necessary.
- Use LaTeX techniques like `\scriptsize`, `\tiny`, `\vspace`, `\hspace`, `\resizebox`, or `minipage` to prevent content overflow.
- Prioritize content readability and visual fit over absolute pixel-perfect replication if the original slide is too dense. Summarize or truncate content if strictly necessary to maintain a clean LaTeX representation that fits the frame.

The overall LaTeX Beamer document structure should be complete, including:
- `\ documentclass{beamer}`
- `\begin{document}`
- Exactly two `\frame` environments (one for each image).
- `\end{document}`

Metadata hints for contextualization (do not include directly in the slides unless relevant to generated content):
```

```
Slide 1 context: <metadata hint 1>
Slide 2 context: <metadata hint 2>

Output ONLY the complete LaTeX Beamer code. Do not include any
    explanations, comments outside the LaTeX code, or additional
    text.
```

Listing 3.11: Prompt used to generate the anonymized slides starting from an initial couple.
NOTE: to allow the document compilation "\" and "documentclass{beamer}" had to be separated using a space while it was not necessary in the original prompt.

3.8.2 Python Post-Processing

After the LaTeX Beamer code was generated by the VLM, we applied a Python post-processing step to ensure that the resulting lides would compile cleanly and render within the spatial constraints of a standard presentation page. While the model produced LaTeX code that was generally syntactically correct, it frequently exhibited layout issues such as overflowing content, oversized diagrams, or overly long bullet lists that disrupted readability and visual balance.

To address these issues, we implemented a set of rule-based transformations aimed at improving layout compactness and preventing visual artifacts. These included: reducing the default font size at the beginning of each slide, wrapping large environments (like tables or bullet lists) in containers that constrained their width, and scaling down diagrams generated with TikZ to avoid clipping. Additionally, redundant preamble declarations—such as multiple occurrences of \texttt\documentclass, were automatically removed to prevent compilation errors.

These adjustments, although heuristic in nature, substantially improved the consistency and reliability of the anonymized slides, enabling large-scale LaTeX generation with minimal manual intervention.

3.8.3 Validation of Anonymized Slides

All generated slides were compiled into `PDF` documents and then converted into individual slide images. These images were then evaluated using `Gemini 2.5 Pro` to validate key quality criteria:

- All content must be clearly visible and legible.
- No graphical or textual elements should overlap.
- Layout should be consistent and presentation-like.

The prompt used for the validation is the following:

```
You are an expert presentation reviewer. I will provide you with two images that represent a two-slide presentation. Your task is to evaluate the quality of this entire two-slide presentation based *solely* on content visibility.

Here are the criteria for a 'correctly formatted' and 'all information visible' presentation:

1.  **Content Visibility (for EACH slide):**
    * Is all text, images, and other graphical elements completely visible on *both* slides?
    * Are there any parts of the content cut off by the slide boundaries (top, bottom, left, right) on *either* slide?
    * Are there any elements overlapping in a way that makes information unreadable on *either* slide?

Based on these detailed criteria, please respond with 'GOOD' if the entire two-slide presentation meets all requirements. If it has ANY issues, respond with 'BAD' and clearly explain the specific problems found, indicating which slide(s) are affected. Be precise.

**Examples of BAD responses:**
* BAD - Slide 1: Text cut off at the bottom right corner.
* BAD - Slide 2: The figures overlap hiding each other.
```

Listing 3.12: Prompt used to validate the anonymized slides.

3.9 Anonymized Slide Extension

Given the anonymized slides, the next step consisted in extending the content of each single slide by prompting a VLM to generate additional slides while maintaining contextual coherence:

1. We first gathered all slide couples that passed the anonymization quality check.
2. Each anonymized slide pair was compiled into a PDF and split into individual slide images.
3. For each singe slide image, a request was built for Gemini 2.5 Pro using a detailed multimodal prompt. This prompt included:
    - The original section title and summary where the anonymized slide comes from
    - A list of all sections in the original report, each with its title and summary

- The anonymized slide as an image
- The desired number of additional slides to generate (typically 10 but can be tuned as a hyperparameter)

4. The LLM generated a complete LaTeX Beamer document extending the input content.

5. The generated code was passed through a post-processing script to ensure formatting and layout correctness

6. The finalized LaTeX documents were then compiled into PDFs.

This procedure allowed us to use the anonymized slides as the seeds for the generation and then extend their context to obtain a more complete presentation.

Here the prompt used to extend the single slide to a presentation:

```
Expand the following anonymized slide into a full LaTeX Beamer
   presentation.

Requirements:
- Maintain consistency of anonymized names throughout (e.g.,
   countries, companies, products).
- Do not invent new anonymized entities; reuse the provided ones.
- Follow this detailed presentation structure for content
   generation:

<formatted_presentation_structure>
[
    {
        "section_title": "Title",
        "start_page_index": 0,
        "end_page_index": 1,
        "summary": "Summary goes here."
    },
    ...
]

- Generate about <slide_count> slides, with 1-2 slides dedicated
   to each of the sections provided.
- Output only the complete LaTeX Beamer code for the slides,
   including the \ documentclass{beamer}, \begin{document}, \end{
   document} and \frame environments. Do not include any
   explanations, comments outside the LaTeX code, or additional
   text.

Anonymized Initial Slide section:

{
```

```
    "section_title": "Source Title",
    "start_page_index": 10,
    "end_page_index": 14,
    "summary": "Source Summary goes here."
},
```

Listing 3.13: Prompt used to extend a single anonymized slide to an anonimyzed presentation.

3.10 Experimental Setup

3.10.1 Datasets

The main employed dataset is a private and provided by a private company and it consisted of 13 presentations for a total of 1148 pages. Of these pages only 976 have been used for the questions production as the other presentations were supplied later in the thesis development. The additional slides were still embedded and provided a more challenging task to the RAG pipeline. To test the basic RAG pipeline on single slides, parts of the Vidore dataset were used, specifically:

- `vidore/syntheticDocQA_artificial_intelligence_test`
- `vidore/syntheticDocQA_energy_test`
- `vidore/syntheticDocQA_government_reports_test`
- `vidore/syntheticDocQA_healthcare_industry_test`

3.10.2 Preprocessing

In the preprocessing all the pdf documents were parsed using pymupdf and all the images and single pages were processed and saved as images with a resolution of 200dpi. The embeddings were saved in chromadb and all the embedding distances were calculated using cosine similarity.

3.10.3 Open-Source LLMs Employed

In the pipeline various Open-Source models, all available in Hugging Face, have been employed in different steps:

- `dunzhang/stella_en_1.5B_v5` was employed as the embedder-retriever in the text-only pipeline.

- `royokong/e5-v` was employed as the embedder-retriever in the separated text-image pipeline.
- `MrLight/dse-qwen2-2b-mrl-v1` was employed as the embedder-retriever in the whole-slide pipeline.
- `Qwen/Qwen2-VL-7B-Instruct` was employed as the images description generator in the text-only pipeline, as the answer, meta-answer and aggregated answer generator in all the answer generation techniques.

3.10.4 Proprietary LLMs Employed

In the development various external proprietary models have been employed in different steps:

- Anthropic Claude 3.5 Sonnet was used in the QA pairs generation for the single-slide case
- Anthropic Claude 4 Sonnet was instead used in the QA pairs generation for the multi-slide case
- Google Gemini 2.5 was exmployed in the slides anonymization and extension procedures. It was also employed in the anonymized slides evaluation.
- OpenAI GPT-4o was exmployed as the evaluator to compare the generated answers with the ground-truth.

3.10.5 Evaluation Metrics

After preliminary tests with classic metrics such as String Presence, BLEU score, ROUGE score, Exact Match and LLM or embeddings based metrics such as LLMContextRecall, FactualCorrectness, SemanticSimilarity and Faithfulness all powered by the RAGAS framework. In the end we adopted the langchain qa evaluator component and relied on OpenAI GPT-4o to express a judgement over the generated answers.

3.10.6 Environment

All the tests where models were locally deployed were executed in Colab using the A100 GPU environment. The other parts where no models were deployed such as the answers evaluation phase or the slides anonymization and extension have been executed either locally or on the High RAM CPU Colab environment.

The python packages requirements are available at the beginning of each employed Jupyter notebook in the GitHub repository: `https://github.com/danielemansillo/multimodal-rag-helper`.

Chapter 4

Results and Discussion

In this chapter, we present and analyze the results of experimental pipeline. We discuss the effectiveness of the proposed methods, and, when applicable, compare them with some related literature works.

4.1 Synthetic QA Dataset for Single-Slides Production

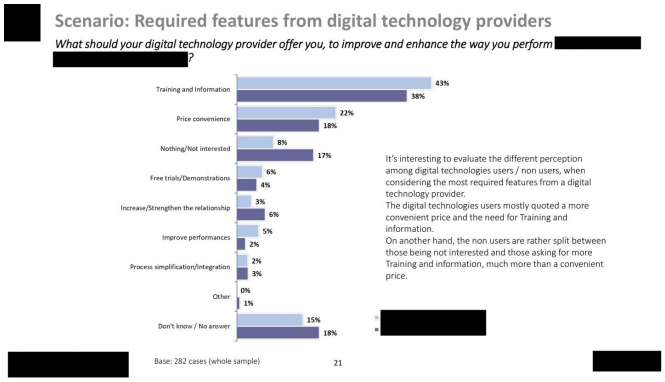
To build a synthetic QA dataset, a total of 976 slides were extracted from the original presentations. For each slide, a factoid-style QA pair was generated using Claude 3.5 Sonnet. Each QA pair includes:

- A **factoid-style question** grounded in the slide's content, when applicable it focus on charts, tables and in general multimodal information.
- A concise, **fact-based answer**.
- An **explanation** grounded in the slide's visual and textual content

A real example of a generated QA pair based on one of the slides is shown in Figure 4.1.

4.2 Embedding and Retrieval techniques comparison

To identify the most effective embedding and retrieval strategy for the QA task, we evaluated three distinct approaches:



Query: What is the highest-priority feature requested from digital technology providers by both users and non-users?

Answer: Training and Information — 43% for [redacted] providers and 38% for non-users of digital technologies.

Explanation: The bar chart shows that "Training and Information" ranks highest for both groups: 43% of [redacted] users and 38% of non-users selected it as their top need. Other features such as price convenience (22% and 18%) and free trials/demos (6% and 4%) trail behind significantly, making Training and Information the clearly dominant priority across user types.

Figure 4.1: Synthetic QA example with associated document image, generated question and answer, and supporting explanation. All sensitive private information has been covered.

- **Text-only embedding**: Visual content was first described using a Vision-Language Model (VLM), and the resulting text was embedded using a text-only embedder.

- **Mixed embedding**: Text and image embeddings were generated separately and concatenated or fused before similarity computation.

- **Whole-slide embedding**: The slides were embedded directly as images without considering textual transcriptions.

Each method was evaluated on the previously generated synthetic QA dataset using Top-1 as the retrieval metric. The retrieval step aimed to determine how accurately each method could return the relevant slide for a given query.

4.2.1 Modality-Specific Retrieval Experiments

To isolate the contribution of different data modalities to retrieval performance, we conducted additional experiments where the input to the retrieval system was deliberately restricted:

- **Image-only:** Queries were matched exclusively against image embeddings extracted from the non-textual visual elements of the slides (excluding full-page).

- **Text-only:** Only the extracted textual content from the slides was used for embedding and retrieval. Visual features or full-slide image embeddings were excluded entirely.

- **Full-page:** The entire slide was treated as a unit of information. Depending on the retrieval technique, this either referred to: a concatenation of slide text and image captions (in the text-only method), or a holistic visual embedding of the full page (in the whole-slide method).

This controlled setup allowed us to have a clearer understanding of how each modality contributed to retrieval effectiveness. Interestingly, as shown in Table 4.1, restricting the input modality sometimes improved performance suggesting that less noisy, more structured representations can outperform dense or overly rich inputs.

Table 4.1: Comparison of Retrieval Techniques by Retrieval Configuration

Embedding Technique	Retrieval Type	Top-1 Accuracy	Top-3 Accuracy	MRR
Mixed	Text	0.243	0.559	0.467
	Image	0.079	0.131	0.121
	All	0.2410	0.518	0.439
Whole slide	Text	0.384	0.620	0.515
	Page	**0.502**	**0.737**	**0.645**
	All	0.427	0.660	0.561
Textual	Text	0.367	0.573	0.485
	Image	0.262	0.397	0.352
	Page	0.347	0.585	0.352
	All	0.366	0.549	0.476

4.2.2 Embedding Time

We also considered the computational cost of each embedding approach. While the time required to embed the slides was comparable across the Mixed and Whole Slide methods (approximately 1.3 seconds per slide for a total of 25 minutes for all slides), the Text-only method required significantly more time primarily due to the need to generate image captions using the VLM.

The image description generation step took approximately 25 seconds per slide for a total of 8 hours for the full dataset. However, once generated, the actual embedding of those captions was fast. This trade-off reflects a fundamental difference between preprocessing overhead and runtime latency: text-based approaches can be more interpretable, but require more initial computation.

4.2.3 Retrieval Results Comparison

The comparative analysis across the three embedding strategies, Text-only, Mixed, and Whole-slide, highlights notable differences in retrieval effectiveness, with the Whole-slide embedding consistently outperforming the other approaches across all evaluation metrics.

As shown in Table 4.1, the Whole-slide approach achieved the highest Top-1 accuracy (0.502), Top-3 accuracy (0.737), and Mean Reciprocal Rank (MRR) of 0.645 when using full-page embeddings. This suggests that treating the entire slide holistically, as a unified visual artifact, results in more semantically aligned representations for the retrieval task in the multimodal QA setting.

Interestingly, even when only the textual content of the slide was used, the Whole-slide embedder still outperformed the Text-only method (Top-1: 0.384 vs. 0.367). This may be attributed to the fact that the Whole-slide model was trained specifically on slide presentations, resulting in embeddings that are more aligned with the structure and semantics of the target task, even when operating solely on text. In contrast, the Text-only embedder lacks this domain-specific alignment.

Another critical observation is that using all modalities together (text, image, and page-level representations) did not consistently yield the best results. In fact, for all embedding techniques, the “All” configuration underperformed compared to the best single-modality configuration. This suggests that the fusion of heterogeneous modalities without careful weighting or alignment may just increase the amount of data to filter and create noise diluting rather than enriching the embeddings database.

4.3 Complete RAG pipeline on synthetic QA dataset

Having identified the optimal embedding and retrieval strategy in the earlier stages, we proceeded to analyze the performance of several answer generation techniques. These techniques were designed to improve the final response quality in a multimodal Retrieval-Augmented Generation (RAG) pipeline, using different strategies for selecting or aggregating retrieved slide content.

4.3.1 Answer Accuracy Comparison

Table 4.2 reports the evaluation results on the synthetic QA dataset, which contains question–answer pairs referencing a single relevant slide. Each technique was assessed on its ability to produce correct answers compared to the ground truth. To have a deeper view on the process we distinguished between cases where the answer was correlated by a correct top-1 match from the retriever and those where it was not.

The Top-1 baseline achieved strong performance with 443 correct answers out of 976 total questions, including 344 cases where the top-1 retrieved slide matched with the one used to produce the qa pair and the answer was correct. This simple method, unsurprisingly achieved the highest number of correct answers correlated by correct top-1 matches, making it an efficient and competitive baseline.

However, several optimization techniques outperformed, even if marginally, the baseline in overall accuracy:

- Technique 3 with Chain of Thought (CoT) achieved the highest number of correct answers (472), outperforming the baseline by a relative 6.55%. By prompting the model to explicitly reason about image relevance before selecting the best candidate slide, it improved both judgment and final output quality. This technique maintained high performance even when the top-1 retrieved slide was not correctly matched, demonstrating effective self-evaluation and reranking. Interestingly the same technique without Chain of Thought prompt achieved the second worst result showing how relevant this simple change can be.

- Technique 4 with normalized distance–perplexity scoring also surpassed the baseline even if just for a total of 5 correct answers, yielding 448 correct answers. We can see how in contrast with other techniques the increment in the correct answers given the wrong top-1 match was very modest hinting that the re-ranking mechanism might be still too cautious.

- Technique 1 (Slide Image + Text Context) and Technique 2 (Progressive Accumulation) performed slightly below the baseline (344 and 426 correct answers, respectively), but were effective in recovering correct answers from lower-ranked slides. Notably, Technique 1 almost doubled the number of cases with a wrong top-1 match but a correct produced answer compared to the baseline (172 vs. 99), showing that its aggregation strategy can compensate for suboptimal retrieval in many cases.

- Technique 4 without normalization performed the worst, highlighting that naive application of perplexity-based re-ranking is insufficient. Its low accuracy suggests that score calibration is essential for such a technique to be effective.

Table 4.2: Comparison of Answering Techniques on internal dataset and synthetic QA

Metric	**Techniques**						
	Top 1	**1**	**2**	**3**	**3 (CoT)**	**4**	**4 (norm.)**
Top-1 Match & Correct	**344**	172	258	152	322	68	342
Wrong Match & Correct	99	**172**	168	131	150	80	106
Total Correct Answer	443	344	426	283	**472**	148	448
Top-1 Match & Wrong	146	318	232	338	168	422	148
Wrong Match & Wrong	387	314	318	355	336	406	380
Total Incorrect Answer	533	632	550	693	504	828	528
Total Top-1 Match	490						
Total not Top-1 Match	486						

4.3.2 Latency–Accuracy Trade-Off

Despite their improved performance, optimization techniques come at a cost. As detailed in Table 3.1, they require multiple LLM invocations per answer.

Thus, a clear trade-off emerges: accuracy might improve with more advanced techniques such as Technique 3 with Chain Of Thought, but latency increases accordingly. For latency-sensitive applications, the Top-1 baseline offers the best compromise, while applications demanding high precision may benefit from the usage of Technique 3 with Chain Of Thought.

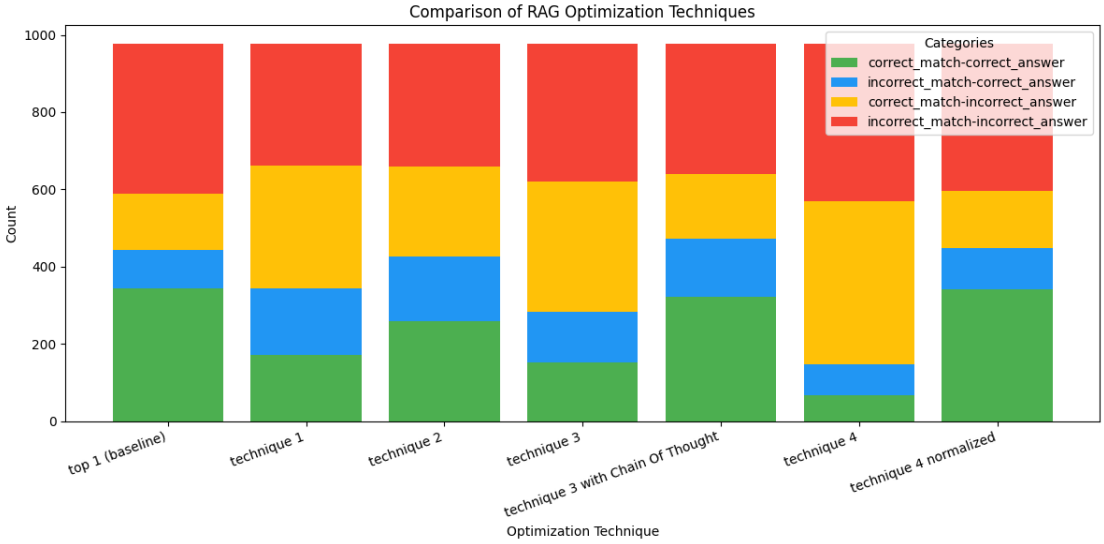


Figure 4.2: Comparison of Answering Techniques on our internal dataset with synthetic QA pairs.

4.4 Complete RAG pipeline on Vidore

To evaluate the proposed RAG pipeline, we extended testing to multiple open-source datasets from the Vidore collection, covering diverse domains such as healthcare, government reports, artificial intelligence, and energy. The format of these datasets does not align perfectly with the scope of the synthetic dataset as they represent multi-page documents more than slide presentations. Each dataset was composed of 1000 pages. The goal was to compare the performance of the four best performing answer generation strategies:

1. Top-1 (baseline)
2. Progressive Accumulation
3. Confidence-Based Selection via VLM-Evaluation with Chain Of Thought
4. Distance + Perplexity-Based Scoring

The results for each dataset are visible in Tables

- `vidore/syntheticDocQA_healthcare_industry_test_results`: Table 4.3
- `vidore/syntheticDocQA_government_reports_test_results`: Table 4.4
- `vidore/syntheticDocQA_artificial_intelligence_test_results`: Table 4.5

- `vidore/syntheticDocQA_energy_test_results`: Table 4.6 aggregated: Table 4.7.

Table 4.3: Comparison of Answering Techniques on `vidore/syntheticDocQA_healthcare_industry_test`

Metric	Techniques			
	Top 1	**2**	**3 (CoT)**	**4 (norm.)**
Top-1 Match & Correct	81	79	**82**	**82**
Wrong Match & Correct	0	**7**	4	0
Total Correct Answer	81	**86**	**86**	82
Top-1 Match & Wrong	10	12	9	9
Wrong Match & Wrong	9	2	5	9
Total Incorrect Answer	19	14	14	18
Total Top-1 Match	91			
Total not Top-1 Match	9			

Table 4.4: Comparison of Answering Techniques on `vidore/syntheticDocQA_government_reports_test`

Metric	Techniques			
	Top 1	**2**	**3 (CoT)**	**4 (norm.)**
Top-1 Match & Correct	84	84	83	**86**
Wrong Match & Correct	2	**5**	4	3
Total Correct Answer	86	**89**	87	**89**
Top-1 Match & Wrong	10	10	11	8
Wrong Match & Wrong	4	1	2	3
Total Incorrect Answer	14	11	13	11
Total Top-1 Match	94			
Total not Top-1 Match	6			

4.4.1 Aggregated Analysis

Unlike in the synthetic QA setting, Technique 2 achieves the best performance on this dataset. It is also the only method among those tested that includes an aggregation step. Its superior results may be attributed to the already high precision of the retriever, which reduces the marginal benefits of Techniques 3 and

Table 4.5: Comparison of Answering Techniques on `vidore/syntheticDocQA_artificial_intelligence_test`

Metric	Techniques			
	Top 1	**2**	**3 (CoT)**	**4 (norm.)**
Top-1 Match & Correct	**83**	82	77	**83**
Wrong Match & Correct	0	**3**	**3**	0
Total Correct Answer	83	**85**	80	83
Top-1 Match & Wrong	11	12	17	11
Wrong Match & Wrong	6	3	3	6
Total Incorrect Answer	17	15	20	17
Total Top-1 Match	94			
Total not Top-1 Match	6			

Table 4.6: Comparison of Answering Techniques on `vidore/syntheticDocQA_energy_test_results`

Metric	Techniques			
	Top 1	**2**	**3 (CoT)**	**4 (norm.)**
Top-1 Match & Correct	**77**	73	71	75
Wrong Match & Correct	1	**6**	5	2
Total Correct Answer	78	**79**	76	77
Top-1 Match & Wrong	10	14	16	12
Wrong Match & Wrong	12	7	8	11
Total Incorrect Answer	22	21	24	23
Total Top-1 Match	87			
Total not Top-1 Match	13			

4, both of which focus primarily on re-ranking. As a result, their contributions may not be substantial enough to outperform the advantages of aggregation.

4.5 Synthetic QA dataset for slide pairs production

To enable cross-document reasoning and support comparative QA tasks, we extended our synthetic dataset by generating factoid questions grounded in pairs of semantically related slides. This process involved a multi-stage pipeline that

Table 4.7: Aggregated Comparison of Answering Techniques on all the vidore Datasets

Metric	**Techniques**			
	Top 1	**2**	**3 (CoT)**	**4 (norm.)**
Top-1 Match & Correct	325	318	313	**326**
Wrong Match & Correct	3	**21**	16	5
Total Correct Answer	328	**339**	329	331
Top-1 Match & Wrong	41	48	53	40
Wrong Match & Wrong	31	13	18	29
Total Incorrect Answer	72	61	71	69
Total Top-1 Match	366			
Total not Top-1 Match	34			

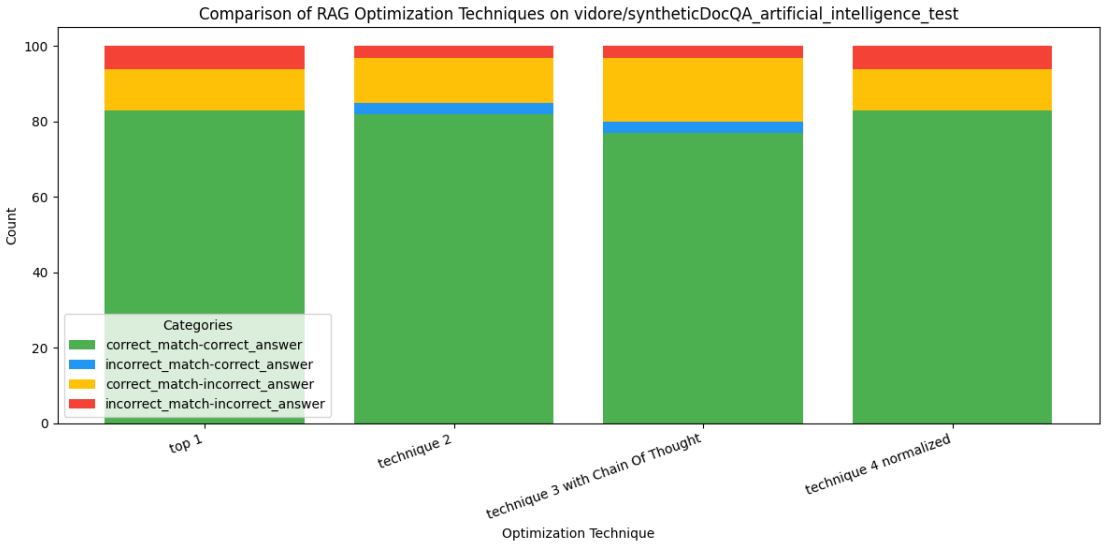


Figure 4.3: Comparison of Answering Techniques on `vidore/syntheticDocQA_artificial_intelligence_test`.

included document sectioning, cross-section similarity analysis, slide-level matching, and final question generation. Below, we describe each step along with its quantitative outcomes and insights.

4.5.1 Presentations Sectioning

The first stage involved dividing each presentation into semantically coherent sections to ensure that slide comparisons would be made between conceptually

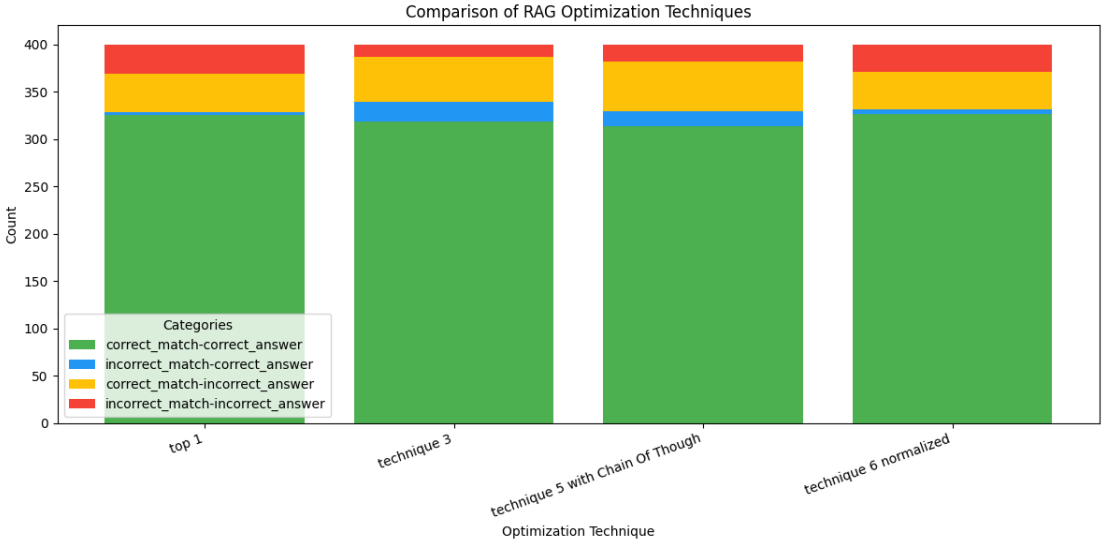


Figure 4.4: Comparison of Answering Techniques on `vidore/syntheticDocQA_energy_test`.

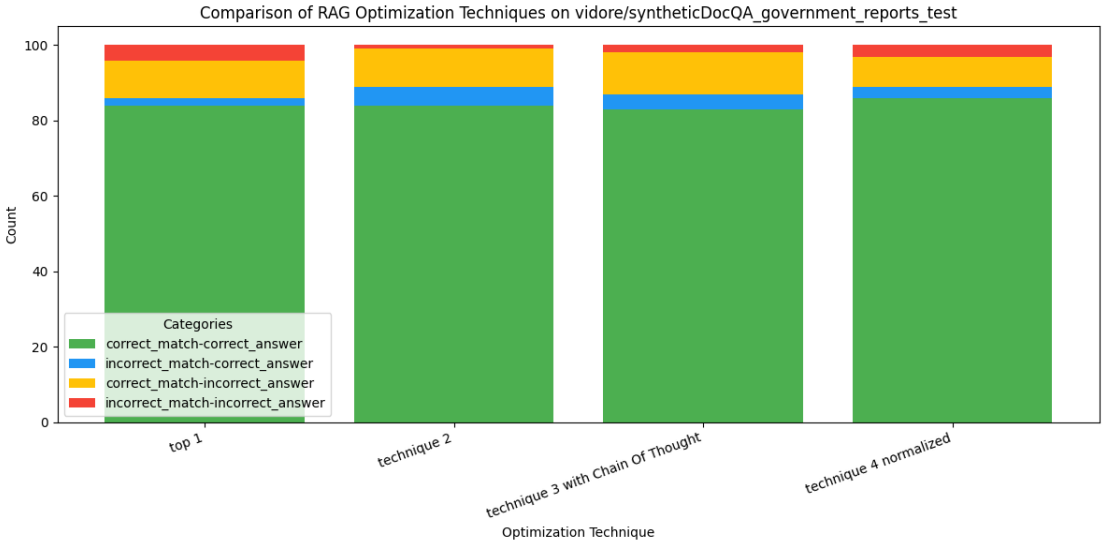


Figure 4.5: Comparison of Answering Techniques on `vidore/syntheticDocQA_government_reports_test`.

aligned sections rather than arbitrary pages. This segmentation was based on the full textual content of each document and performed using Claude 3.5 Sonnet, a large language model capable of high-level document understanding.

This produced a total of **99 sections** across **11 presentations**, with an average

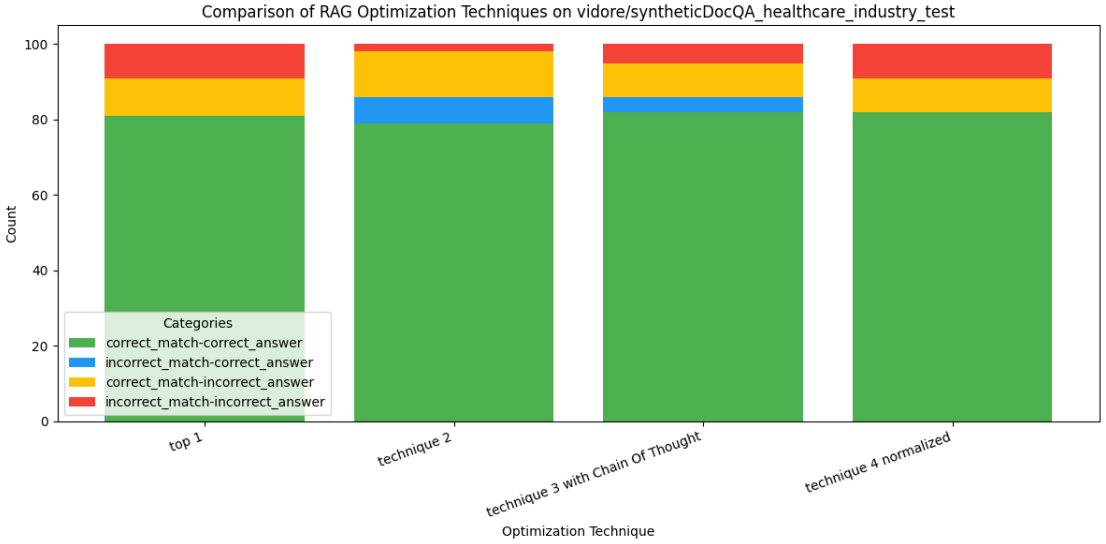


Figure 4.6: Comparison of Answering Techniques on `vidore/syntheticDocQA_healthcare_industry_test`.

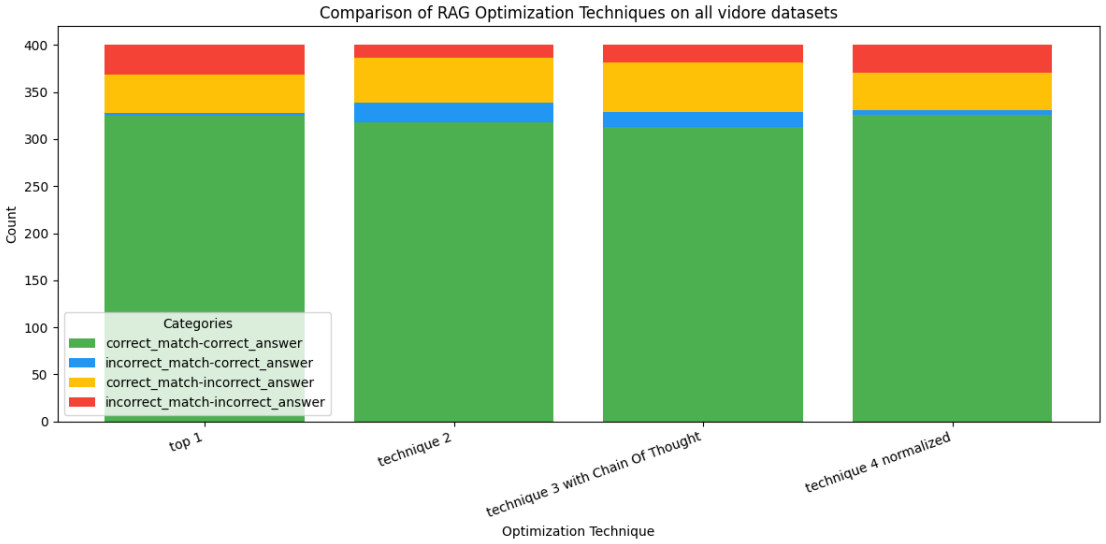


Figure 4.7: Comparison of Answering Techniques on all vidore datasets.

of 9 sections per presentation. The structure of the documents were very similar usually consisting of 3 introduction sections followed by a variable number of core sections and one or two for the conclusion.

Here an example of sectioned presentation:

```
{'path': '/path/to/presentation.pdf',
```

```
'name': 'presentation.pdf',
'country': 'France, Italy, Spain',
'year': 2025,
'sections_text': [{'section_title': 'Methodology',
  'start_page_index': 1,
  'end_page_index': 3,
  'summary': 'Outlines the research methodology for the
  quantitative study ... purchasers.'},
 {'section_title': 'Executive Summary',
  'start_page_index': 4,
  'end_page_index': 6,
  'summary': 'Provides key findings on sample segmentation
  ... selection.'},
 {'section_title': 'SEGMENTATION',
  'start_page_index': 7,
  'end_page_index': 11,
  'summary': 'Details segmentation parameters including
  demographics, ... with varying levels of specialization
  across countries.'},
 {'section_title': 'IOS Penetration',
  'start_page_index': 12,
  'end_page_index': 15,
  'summary': 'Analyzes the penetration of ... preferences
  across countries and age groups.'},
 {'section_title': 'Factors supporting IOS choice',
  'start_page_index': 16,
  'end_page_index': 21,
  'summary': 'Identifies key decision factors when choosing
   ... effectiveness.'},
 {'section_title': 'IOS: limits and barriers for non owners
   only',
  'start_page_index': 22,
  'end_page_index': 26,
  'summary': 'Examines barriers preventing non-owners from
  adopting ... compatibility issues, and full arch concerns
  .'}]}
```

Listing 4.1: Example of extracted sections

This sectioning step was crucial to structure the documents into thematic units. These units then enabled more targeted similarity computations. It allowed the alignment of slides not only based on superficial visual resemblance, but also on the treated topic and on their role inside the presentation, improving the relevance of paired slides across presentations. Manual inspection confirmed that the generated summaries were generally accurate and coherent with the actual content of the

sections.

4.5.2 Sections Pairing

To identify semantically related content across documents, each section was embedded using the `all-MiniLM-L6-v2` sentence transformer. The embedding was computed by concatenating each section's title and summary. We then calculated cosine similarity between all section pairs from different presentations.

A similarity threshold of **0.9** was selected to retain only highly relevant section pairs. This yielded a total of **7 high-confidence section pairs**. The threshold was chosen to balance precision and recall; while lowering it to 0.8 increased the number of pairs to 39, it also introduced semantically weaker matches. For our purpose of generating meaningful comparative questions, a stricter threshold proved more effective.

4.5.3 Slides Similarity search

Using the 7 section pairs identified in the previous step, we conducted a fine-grained similarity search at the slide level. Specifically, all slide combinations within each section pair were compared using visual embeddings. A similarity band of [**0.7, 0.9**] was applied to filter out both dissimilar pairs (noise) and near-identical slides (e.g., repeated intros or disclaimers).

From 1018 candidate slide pairs, this filtering process resulted in **192 slide pairs**, which were deemed visually similar yet semantically distinct enough to support interesting comparative questions.

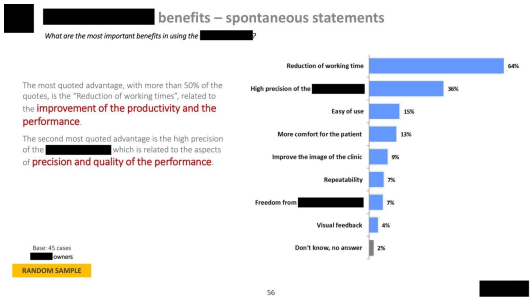
This step confirmed that high-level section similarity is a useful proxy for identifying informative slide pairings, and that visual embeddings can effectively filter out redundant visual content.

4.5.4 Questions Generation

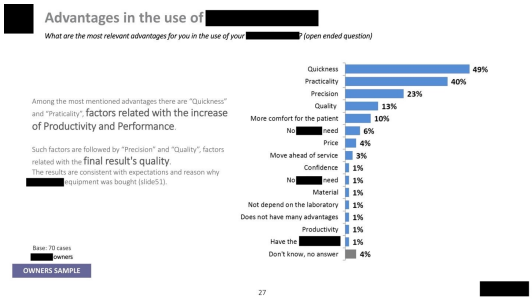
Each of the 192 slide pairs was then passed to Claude 3.5 Sonnet to generate a factoid question and answer with an accompanying explanation. The questions were designed to highlight differences or extract insights and relationships from both slides.

Due to occasional errors in generation such as malformed output a total of **175** valid questions were successfully produced. This represents a generation success rate of approximately **91.1%**.

Here an example of a question produced from a pair of similar slides:



Slide 1 (Italy 2017)



Slide 2 (Brazil 2022)

```
{
 'Factoid question': 'What is the difference in percentage
   points between the top-ranked XXXXXXXXXXXXXXX benefit in
    Italy 2017 and Brazil 2022?',
 'Answer': '15 percentage points',
 'Explanation': "In Italy 2017, the top benefit was '
   Reduction of working time' at 64%, while in Brazil 2022,
   the top benefit was 'Quickness' at 49%. The difference is
    64% - 49% = 15 percentage points."
}
```

Listing 4.2: Example of produced question for slide pairs (private information has been hidden).

4.5.5 Discussion

This pipeline demonstrated that combining semantic sectioning with multimodal similarity search enables scalable generation of comparative QA data. A key insight is that filtering slide pairs based on both section similarity and visual difference helps produce questions that are neither too obvious nor too vague.

Moreover, our choice of thresholds at each step proved effective in balancing dataset quality and size. However, future iterations could improve generation robustness by:

- Implementing fallbacks or retries for failed generations;
- Fine-tune the similarity thresholds hyperparameters for optimal balance between size and quality;
- Human-evaluating a subset to assess factual correctness.

These findings suggest that semi-automated pipelines for cross-document QA generation are not only feasible but can yield high-quality examples for downstream retrieval or reasoning tasks.

4.6 End-to-end RAG pipeline Evaluation on Synthetic Multi-Slide QA Dataset

After generating the synthetic QA pairs for slide pairs, we evaluated the performance of the complete retrieval-augmented generation (RAG) pipeline in a multi-slide context. This evaluation is based on the 175 QA pairs described in the previous section. The pipeline was adapted to accommodate multi-slide reasoning, as outlined in Section 3.

4.6.1 Evaluation Methodology

To assess performance, we applied the following procedure for each QA pair:

1. The question was submitted to the retriever.
2. The top-n most relevant slides (n=10) were retrieved.
3. Depending on the employed generation strategy, a subset of retrieved slides was selected and processed.
4. A final answer was generated based on the selected context.
5. The generated answer was compared to the reference answer using the langchain QA LLM evaluator component powered by GPT-4o.

Table 4.8 summarizes the retrieval performance, capturing how often the original slides from the QA pair appeared in the top-n retrieved results.

Table 4.8: Top-k Retrieval Accuracy for Slide Pair QA Dataset

Metric	**Top-1**	**Top-2**	**Top-3**	**Top-5**	**Top-10**
Match Image 1	25.71%	36.00%	41.14%	50.86%	59.43%
Match Image 2	12.00%	18.86%	25.71%	33.14%	45.71%
Any Match	37.71%	51.43%	61.71%	73.71%	83.43%
Both Match	–	3.43%	5.14%	10.29%	21.71%

4.6.2 Answer Generation Results

The results of each of these techniques are available in the chart 4.8 below:

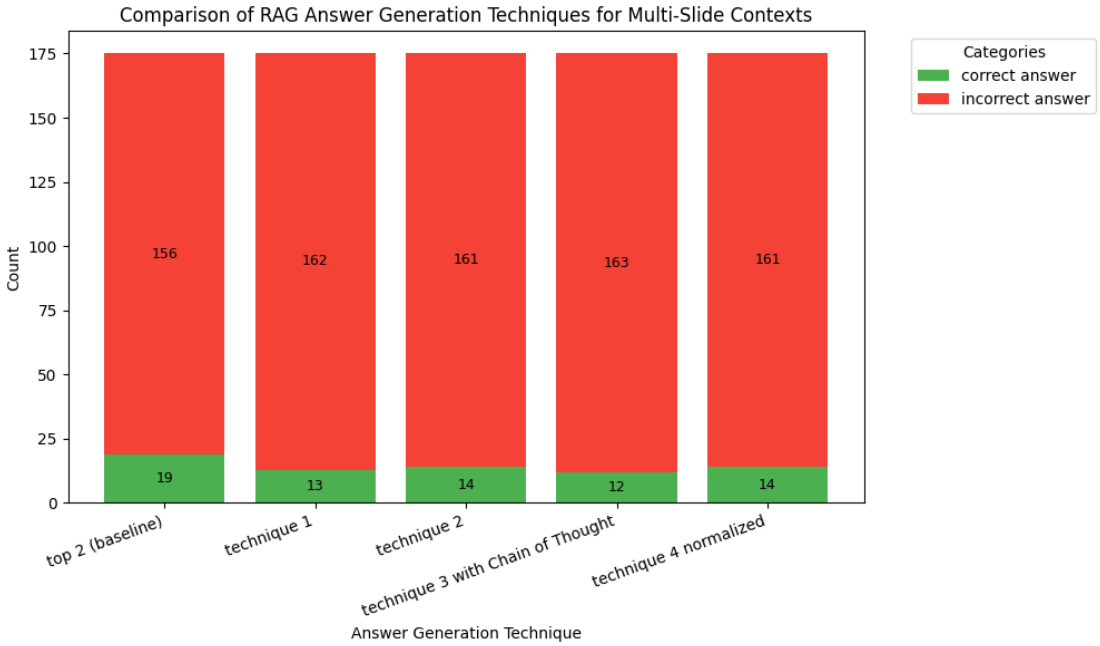


Figure 4.8: Comparison of Answering Techniques on our internal dataset with synthetic multi-slide QA pairs.

and in the following table 4.9:

Table 4.9: Answer Accuracy for Different Techniques for Slide Pair QA Dataset

Technique	Correct	Incorrect	Accuracy
Top 2 (baseline)	19	156	10.85%
Technique 2	13	162	7.42%
Technique 3 with COT	14	161	8.00%
Technique 4	12	163	6.85%
Technique 4 normalized	14	161	8.00%

4.6.3 Discussion

The results highlight several key challenges specific to the multi-slide QA scenario:

Retrieval Performance as a Bottleneck The primary limitation lies in the difficulty of retrieving both target slides from the original QA pair. For instance, while at least one relevant slide appears in the top-2 results for over 51% of questions, both slides are retrieved in only 3.43% of cases. Even at top-10, this number remains relatively low at 21.71%. This significantly constrains the context available for generation, resulting in reduced answer quality across all evaluated techniques.

Answering Performance and Strategy Comparison Among all tested techniques, the simple Top-2 baseline achieved the highest accuracy (10.85%), although still remarkably lower than in the single-slide scenario. Notably, none of the optimization strategies designed for multi-slide reasoning surpassed this baseline. This suggests that in the absence of reliable retrieval, improvements in generation logic yield limited benefits.

Redundancy and Latent Knowledge in the Corpus Interestingly, the gap between low retrieval accuracy for both target slides (3.43%) and relatively higher answer correctness (10.85%) suggests the presence of redundant or overlapping content across presentations. This indicates that some questions can still be answered correctly even when the original slide pair is not retrieved, highlighting the possibility of semantic redundancy within slide decks, an expected characteristic in slide presentations as their main goal is not only to carry information but to present it to a audience and often presenting the same information in different ways can enhance the message comprehension.

First Slide Retrieval Bias It is interesting to notice a consistent gap in retrieval performance between Slide 1 and Slide 2 (around 15%) across all top-k values suggests a possible bias introduced during question generation. The generative model may have implicitly prioritized content from the first slide in the pair, leading to easier retrievability or higher textual alignment with the query.

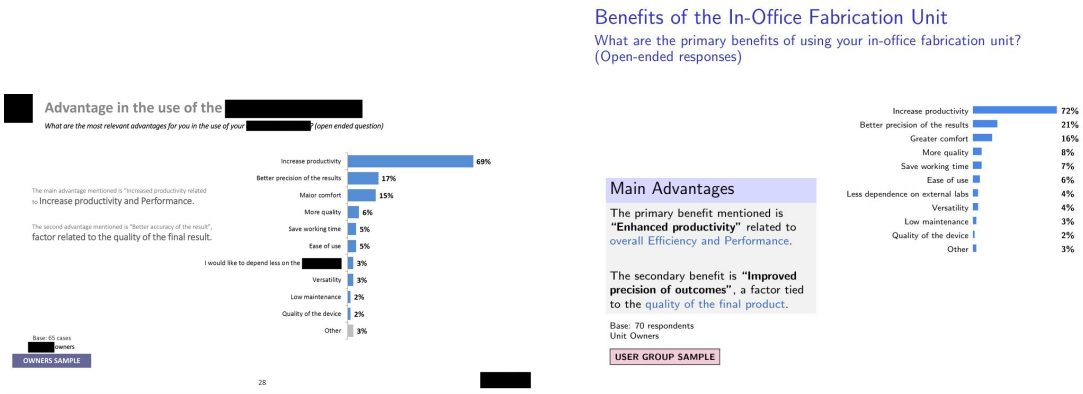
Need for Improved Retrieval Strategies Overall, the multi-slide QA setting reveals the limits of naive embedding-based retrieval when dealing with compound queries requiring multiple discrete evidence sources. Many questions are based on slide pairs that are apparently disconnected, belonging to different documents and referring to different countries and years. In these cases, retrieving both slides requires not only a surface-level similarity but also a deep comprehension of the query intents and of its single requested components. This underlines the need for advanced retrieval architectures such as:

- Multi-hop retrieval

- Query Decomposition;
- Entity-aware re-ranking.

4.7 Slides anonymization

The anonymization pipeline proved effective in generating realistic, shareable slide content from confidential or proprietary presentations. Transforming the pairs instead of the single slides allowed to preserve the thematic coherence between the slides which will facilitate the eventual downstream task of question-answer pairs generation on multiple slides. In Figure 4.9 it is possible to see an example of successful anonymization where the original layout was maintained but the content, the context and the entities were shifted.



Left: Original slide Right: Anonymized slide

Figure 4.9: Example of the source and output slides for the anonymization process. All the sensitive information in the original slide has been covered

4.7.1 Impact of Pair-Based Anonymization

We observed that anonymizing slide pairs led to more contextually rich and coherent outputs. Unlike single-slide anonymization, which often caused a strong semantic drift or generated content detached from the original presentation's flow, operating on slide pairs helped maintain inter-slide relationships. This was reflected in higher consistency between slides and more natural transitions in topics, which are essential for use in retrieval and generation tasks.

Additionally, slide pair processing enhanced variability: the visual language model (VLM) treated slide pairs as a more complex input signal, yielding greater

diversity in the anonymized content as visible in Figure 4.10 where a single source slide was transformed into 7 different anonymized slides all with different layouts and contents.

4.7.2 Generation and Post-Processing

The LaTeX Beamer code generated by the VLM was generally well-formed but not without issues. A small but consistent fraction of the slides initially suffered from formatting problems such as text or tables overflowing the slide boundaries or TikZ figures being too large. These issues were effectively mitigated by the Python-based post-processing step, which automatically adjusted font sizes, resized complex elements, and removed redundant LaTeX preamble declarations.

While the post-processing was not flawless, it was fundamental to allow the LaTeX code to compile since none of the produced code for the LaTeX Beamer slides was suitable for compilation right out of the LLM. With this post-processing in place we were able to successfully compile 104 slide pairs out of 175 with a conversion rate of 59.4%. This conversion rate can for sure be improved by fine-tuning the Python post-processing to be more robust and identify and solve even more common issues.

4.7.3 Validation Outcomes

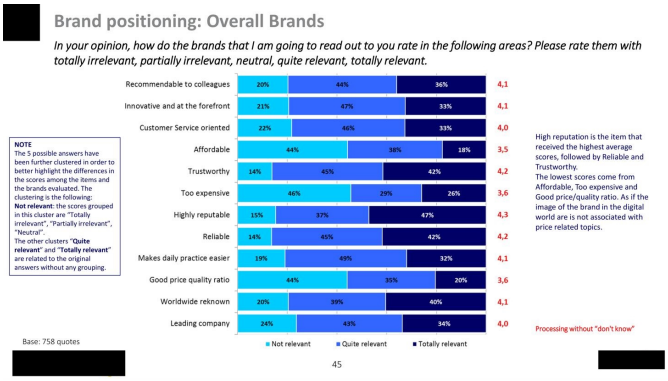
We applied the validation prompt 3.13 to all generated slide images using Gemini 2.5 Pro. This allowed us to simulate an external assessment of slide quality and ensure that the anonymized output met usability standards. The validation focused strictly on content visibility and layout integrity.

Out of the 104 compiled slide pairs 39 out of 104 were rated as "GOOD" by the model, meaning they contained no visual defects, no overlapping content, and no cutoff sections. The remaining 65 typically involved edge cases where charts or text were cut off because they were too big or too long. Some examples of pairs labeled as "BAD" with relative explanation are visible in figure 4.11. This likely happens because the default font employed in LaTeX is bigger than the font used in the original slides resulting in an overflow.

4.7.4 Limitations and Future Improvements

While the anonymization pipeline was effective for our purposes, several limitations remain. The reliance on heuristic post-processing rules means that edge-case layout bugs are not always caught.

In future work, incorporating a few-shot prompting approach for the anonymized slides generation could drastically improve the conversion ratio and reduce the



Source anonymized slide.

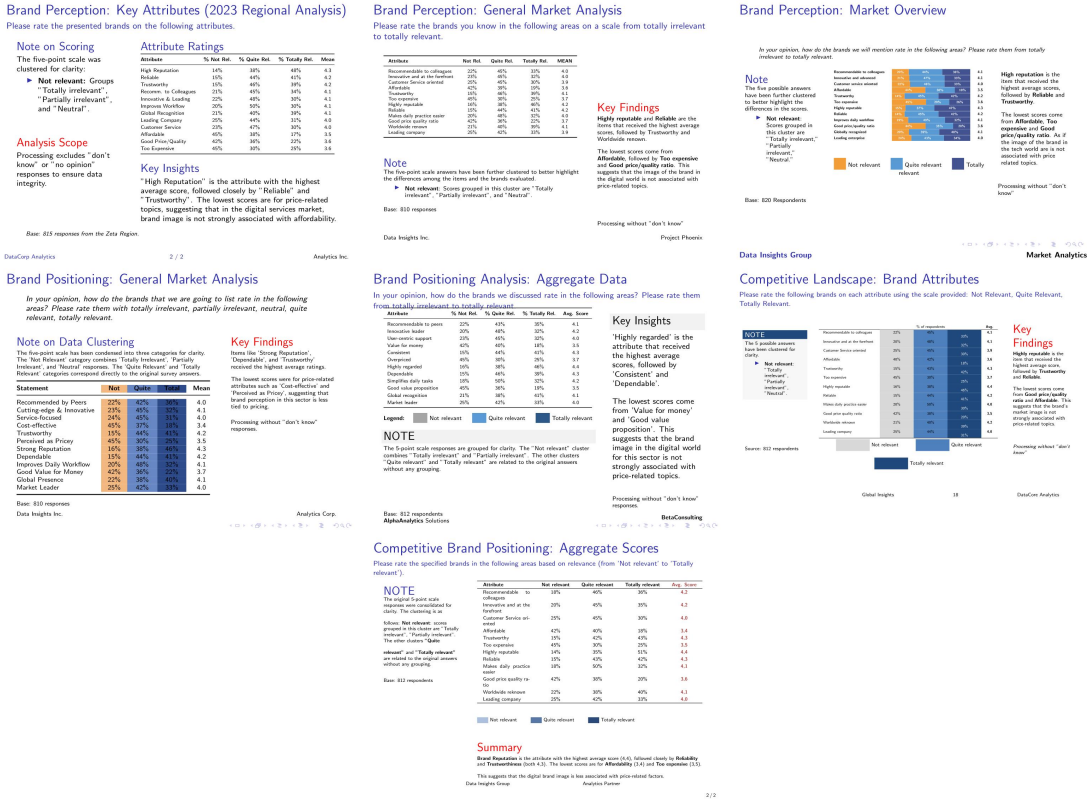
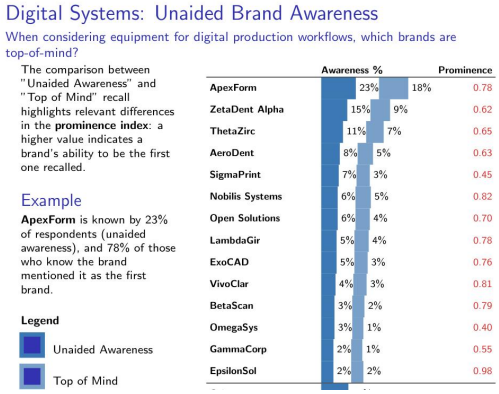
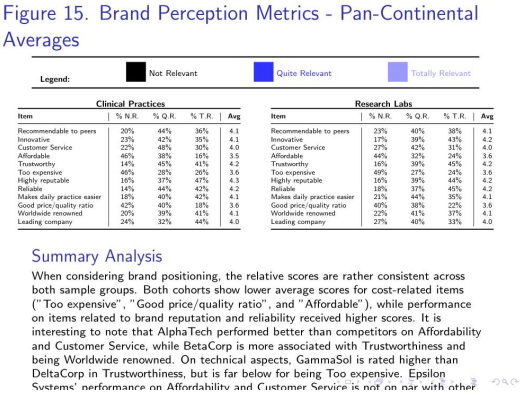


Figure 4.10: Example of how the same source slide can change in different anonymized pairs.

necessity for ad-hoc post-processing procedures. Additionally, fine-tuning a VLM with specific anonymization examples may further increase quality and also reduce



SLIDE 2: THE TABLE/CHART IS CUT OFF AT THE BOTTOM, WITH INFORMATION FOR AT LEAST ONE BRAND (E.G., GAMMACORP AND THE ENTRY BELOW EPSILONSOL) NOT FULLY VISIBLE.')



SLIDE 1: TEXT CUT OFF AT THE BOTTOM IN THE "SUMMARY ANALYSIS" SECTION.

Figure 4.11: Examples of anonymized slides evaluated as "BAD" with provided reason

reliance on post-generation fixes. On the other side improving the Python post-processing by incorporating more fixes for common edge-cases while maintaining the current setup could also improve the general evaluation of the generated slides by avoiding content overlap and overflow.

4.8 Anonymized slides extension

Out of 39 slide pairs that were evaluated as good by a VLM corresponding to a total of 78 single slides, 45 anonymized presentations were successfully generated. The remaining 33 slides could not be compiled into valid presentations. With improvements to the post-processing functions in Python, it is likely that an even greater number of presentations could be successfully obtained.

4.8.1 Presentation Quality and Variability

Empirical evaluation of the generated presentations revealed a noticeable variation in output quality. As illustrated in Figures 4.13 and 4.12, some presentations were visually coherent, well-structured, and maintained a professional layout. Others, however, exhibited several defect such as empty slides, placeholders used instead of

images, or quite empty content.

Despite these inconsistencies, the overall quality tended to be satisfactory. The majority of successful presentations adhered to the desired structural flow, preserved the anonymized context, and maintained the visual conventions expected in formal slide decks. Unfortunately though, the constraint of reusing anonymized entities (e.g., fictional country names) was respected but sometimes labels and entities changed across slides from the same presentation, limiting consistency throughout the generated decks.

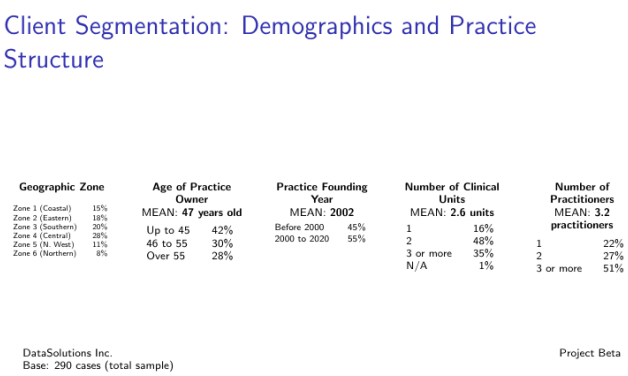
4.8.2 Observations on VLM Behavior

A noteworthy observation emerged regarding the model's ability to infer high-level structure from limited input. Given just a single anonymized slide and contextual metadata, the VLM was able to extrapolate a plausible section-level narrative and distribute it across a new presentation. This behavior supports the viability of using VLMs for complex content generation, especially when grounded in structured prompts.

That said, the model sometimes struggled with consistency across the presentation, highlighting the need for an explicit prompt with entities mappings or prompt tuning to maintain consistency and information richness over longer sequences.

4.8.3 Limitations and Opportunities

One of the main limitations of the current approach lies in the fragility of LaTeX-based rendering. Even minor syntax or formatting inconsistencies can lead to compilation failures, and although our post-processing mitigated many of these, a non-negligible number of cases still failed. Future iterations could explore the possibility of a multi-iteration process where for each failed compilation the error with the source code is passed to the LLM for correction or, as already proposed for the Slides Anonymization process, adopt a few-shot prompting approach to supply the VLM with successful examples on which to build the new representations.

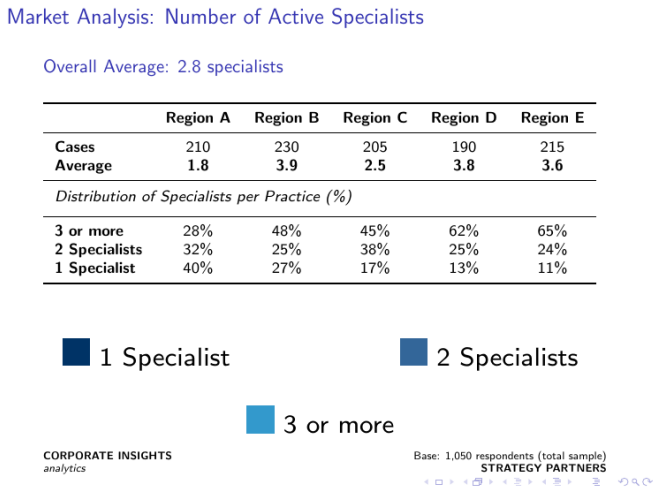


Source anonymized slide.

↓

Target anonymized presentation.

Figure 4.12: Example of an unsatisfactory outcome from the slides anonymization extension process.



Source anonymized slide.

↓

Target anonymized presentation.

Figure 4.13: Example of a good outcome from the slides anonymization extension process.

Chapter 5

Conclusion

This work presented a comprehensive development and evaluation of a complete Retrieval-Augmented Generation (RAG) pipeline for question answering on multimodal slides presentation, first on the single-slide use-case and then extended to the more realistic multi-slide use-case. The study included both controlled experiments on internally developed synthetic QA datasets and external validation on several open-source benchmarks from the `vidore` collection. In the end, to facilitate the expansion of this domain we proposed an anonymization technique that leverages powerful VLMs to mask entities and values while maintaining the layout and overall coherence.

5.1 Summary of Contributions

This work introduces several key contributions toward advancing retrieval-augmented question answering over multimodal slide presentations. First, we developed a synthetic QA dataset specifically designed for the single-slide use-case, enabling controlled evaluation of retrieval and answering strategies. We then conducted an extensive analysis of multiple embedding techniques for retrieval, ultimately identifying whole-slide visual embeddings as the most effective in terms of performance and efficiency. Building on these findings, we designed and implemented a complete Retrieval-Augmented Generation (RAG) pipeline tailored to multimodal slide inputs. Within this pipeline, we explored and compared various advanced answering strategies, such as re-ranking, aggregation, and hybrid scoring mechanisms combining text perplexity and retrieval similarity, to determine their effectiveness. The pipeline's robustness was further validated using external open-source benchmarks from the `vidore` collection. To extend the QA capabilities to more realistic, multi-slide scenarios, we developed a novel approach for generating multi-slide QA data by synthesizing questions based on slide pairs. Additionally, we proposed an

anonymization framework leveraging Vision-Language Models (VLMs) to mask entities while maintaining layout and semantic coherence. This was complemented by a context-aware pipeline to expand anonymized slide pairs into coherent full presentations, enabling scalable production and sharing of privacy-preserving training data for future QA applications.

5.2 Embedding and Retrieval Insights

The initial step in the pipeline construction was the selection of the embedding and retrieval strategy. Three embedding approaches were compared: text-only embeddings derived from Vision-Language Model (VLM) captions, mixed modality embeddings, and whole-slide visual embeddings. Extensive experiments across these configurations revealed that the whole-slide approach, particularly with only page-level embeddings, offered the best retrieval performance, achieving the highest Top-1 accuracy (0.502), Top-3 accuracy (0.737), and MRR (0.645). Surprisingly, simply combining all modalities did not improve performance, suggesting that indiscriminate fusion of heterogeneous signals can introduce noise rather than improve retrieval.

The evaluation also considered computational cost. The text-only embedding strategy incurred a significant preprocessing overhead due to the time-intensive image captioning step. In contrast, both mixed and whole-slide strategies provided a much faster overall pipeline with generally comparable or superior retrieval accuracy.

5.3 Findings from the Evaluation of Answering Techniques

In total 6 answering techniques, excluding the baseline, were compared on the internal synthetic QA dataset and successively the 3 best performing ones were compared on the vidore datasets. These answering techniques ranged from simple answering based on the top-1 retrieval (baseline) to methods involving intermediate answers and retrieved data aggregation, re-ranking using chain-of-thought (CoT) prompting and composed metrics based on text perplexity and retrieval similarity score. Results demonstrate that performance is dependent on dataset characteristics. On the internal synthetic dataset, where the top-1 retrieval accuracy was around 50%, re-ranking approaches delivered better accuracy. Conversely, on Vidore datasets where the top-1 retrieval accuracy was around 90%, the aggregation-based technique consistently outperformed others. This indicates that aggregation is particularly effective when initial retrieval accuracy is already high, whereas re-ranking proves more beneficial in scenarios with noisier retrievals.

5.3.1 Multi-image Dataset Production and Real-World implications

The shift from single-slide to multi-slide QA introduced additional challenges. As no suitable datasets was available, a novel multi-image QA dataset was constructed using synthetic generation techniques. Adapting the pipeline to this setting revealed significant limitations, particularly in the retrieval performance. While retrieving one relevant slide from a pair was common, identifying both consistently proved more challenging, severely impacting downstream QA performance.

These findings underscore that multi-slide QA cannot rely on naive extensions of single-slide methods. Instead, it demands new approaches mostly to overcome the bottleneck in the retrieval phase.

5.3.2 Anonymization Process Outcomes

To facilitate dataset sharing and privacy preservation, an automated pipeline was built to anonymize and expand anomyized slides. This system successfully masks sensitive content while retaining the layout and thematic structure. Furthermore, anonymized slide pairs were extended into coherent multi-slide presentations using context-aware generation.

Despite encouraging results, the pipeline is not flawless. Some generated LaTeX files failed to compile due to errors in the LaTeX code while others presented formatting errors such as overlapping or cut text and context. However, the Python post-processing module significantly mitigated these issues.

5.3.3 Limitations and Challenges

Several limitations were identified throughout this work:

- **Multi-slide Retrieval Bottleneck:** While retrieval of individual slides is reasonably effective, jointly retrieving both relevant slides remains challenging. This limitation critically affects the overall performance of the RAG pipeline in the multi-slide setting.

- **Retrieval-Generalization Trade-offs:** Strategies optimized for single-slide performance do not generalize well to multi-slide retrieval, indicating a need for specialized designs tailored to inter-slide reasoning.

- **Pipeline Sensitivity to Retrieval Accuracy:** Answering performance remains tightly bound to retrieval quality. Low recall in the first stage constrains the upper bound of achievable QA performance.

- **Fragility of LaTeX-based Anonymization:** The anonymization and expansion pipeline, while innovative, occasionally produces .tex files that cannot be compiled or malformed LaTeX documents. Though current post-processing fixes many issues, more robust generation methods are required for production-scale deployment.

5.3.4 Future Work

This project opens up several promising directions for future research:

- **Improved Multi-Slide Retrieval:** Explore entity-aware query decomposition to improve the retrieval of related slide pairs. Another proposal might involve innovative techniques of query expansion. Other proposals might involve enriching the single embeddings with contextual metadata that now is missing or using the extracted entities in the query to query in the vector database metadata to reduce the corpus to query.

- **Improved Answering Module tailored for Multi-Source Input:** Develop answering techniques that natively and efficiently work with multiple slides.

- **Answering Module Evaluation Decoupled from Retrieval:** Benchmark answering techniques using oracle-retrieved slides to isolate the contribution of generation methods alone and better guide design choices.

- **Extend Answering Module Evaluation on Open Datasets:** Test answering techniques using open multi-image datasets.

- **Scalable Anonymized Dataset Production:** Expand the anonymized presentation corpus using more data and more robust generation and validation techniques, facilitating the creation of the first large-scale multi-slide QA benchmark.

- **Explore Optimization techniques for Slides Anonymization:** Employ few-shot prompting or model fine-tuning for slides anonymization. Fine-tuning might yield better results but requires a specialized dataset and potentially expensive training.

- **Automated Post-Processing and Repair:** Extend the current Python post-processing pipeline using common detected errors to automatically fix additional LaTeX errors.

Final Remarks

Overall, this work delivers a complete and extensible framework for RAG pipelines over multimodal slide presentations, alongside tools for scalable dataset creation through anonymization and expansion. It not only addresses a largely underexplored domain but also lays the foundation for future advances in privacy-preserving, multimodal document understanding.

Bibliography

[1] OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774 (cit. on pp. 5, 13).

[2] Gemini Team et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2025. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805 (cit. on pp. 5, 14).

[3] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL]. URL: https://arxiv.org/abs/2307.09288 (cit. on p. 5).

[4] KAREN SPARCK JONES. «A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL». In: *Journal of Documentation* 28.1 (2025/06/23 1972), pp. 11–21 (cit. on p. 5).

[5] G. Salton, A. Wong, and C. S. Yang. «A vector space model for automatic indexing». In: *Commun. ACM* 18.11 (Nov. 1975), pp. 613–620. ISSN: 0001-0782. DOI: 10.1145/361219.361220. URL: https://doi.org/10.1145/361219.361220 (cit. on p. 5).

[6] Stephen Robertson and Hugo Zaragoza. «The Probabilistic Relevance Framework: BM25 and Beyond». In: *Found. Trends Inf. Retr.* 3.4 (Apr. 2009), pp. 333–389. ISSN: 1554-0669. DOI: 10.1561/1500000019. URL: https://doi.org/10.1561/1500000019 (cit. on p. 6).

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL]. URL: https://arxiv.org/abs/1301.3781 (cit. on p. 6).

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL]. URL: https://arxiv.org/abs/1301.3781 (cit. on p. 6).

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805 (cit. on p. 7).

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on pp. 7, 12).

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on p. 8).

[12] Omar Khattab and Matei Zaharia. *ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT*. 2020. arXiv: 2004.12832 [cs.IR]. URL: https://arxiv.org/abs/2004.12832 (cit. on p. 9).

[13] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. *SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking*. 2021. arXiv: 2107.05720 [cs.IR]. URL: https://arxiv.org/abs/2107.05720 (cit. on p. 9).

[14] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020 (cit. on pp. 10, 13).

[15] Lu Yuan et al. *Florence: A New Foundation Model for Computer Vision*. 2021. arXiv: 2111.11432 [cs.CV]. URL: https://arxiv.org/abs/2111.11432 (cit. on p. 10).

[16] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*. 2022. arXiv: 2201.12086 [cs.CV]. URL: https://arxiv.org/abs/2201.12086 (cit. on p. 10).

[17] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models*. 2023. arXiv: 2301.12597 [cs.CV]. URL: https://arxiv.org/abs/2301.12597 (cit. on p. 10).

[18] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. «LayoutLM: Pre-training of Text and Layout for Document Image Understanding». In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery; Data Mining*. KDD '20. ACM, Aug. 2020, pp. 1192–1200. DOI: 10.1145/3394486.3403172. URL: http://dx.doi.org/10.1145/3394486.3403172 (cit. on p. 10).

[19] Yang Xu et al. *LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding*. 2022. arXiv: 2012.14740 [cs.CL]. URL: https://arxiv.org/abs/2012.14740 (cit. on p. 10).

[20] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. *LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking*. 2022. arXiv: 2204.08387 [cs.CL]. URL: https://arxiv.org/abs/2204.08387 (cit. on p. 10).

[21] Xueguang Ma, Sheng-Chieh Lin, Minghan Li, Wenhu Chen, and Jimmy Lin. *Unifying Multimodal Retrieval via Document Screenshot Embedding*. 2024. arXiv: 2406.11251 [cs.IR]. URL: https://arxiv.org/abs/2406.11251 (cit. on p. 11).

[22] Marah Abdin et al. *Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone*. 2024. arXiv: 2404.14219 [cs.CL]. URL: https://arxiv.org/abs/2404.14219 (cit. on p. 11).

[23] Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. *ColPali: Efficient Document Retrieval with Vision Language Models*. 2025. arXiv: 2407.01449 [cs.IR]. URL: https://arxiv.org/abs/2407.01449 (cit. on pp. 11, 22).

[24] Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. *E5-V: Universal Embeddings with Multimodal Large Language Models*. 2024. arXiv: 2407.12580 [cs.CL]. URL: https://arxiv.org/abs/2407.12580 (cit. on p. 12).

[25] Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. *Jasper and Stella: distillation of SOTA embedding models*. 2025. arXiv: 2412.19048 [cs.IR]. URL: https://arxiv.org/abs/2412.19048 (cit. on p. 12).

[26] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. *MTEB: Massive Text Embedding Benchmark*. 2023. arXiv: 2210.07316 [cs.CL]. URL: https://arxiv.org/abs/2210.07316 (cit. on p. 12).

[27] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. «Improving language understanding by generative pre-training». In: (2018) (cit. on p. 13).

[28] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: https://arxiv.org/abs/2001.08361 (cit. on p. 13).

[29] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL]. URL: https://arxiv.org/abs/2005.14165 (cit. on p. 13).

[30] Aakanksha Chowdhery et al. *PaLM: Scaling Language Modeling with Pathways*. 2022. arXiv: 2204.02311 [cs.CL]. URL: https://arxiv.org/abs/2204.02311 (cit. on p. 13).

[31] Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. *DocFormer: End-to-End Transformer for Document Understanding*. 2021. arXiv: 2106.11539 [cs.CV]. URL: https://arxiv.org/abs/2106.11539 (cit. on p. 13).

[32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485 [cs.CV]. URL: https://arxiv.org/abs/2304.08485 (cit. on p. 14).

[33] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. *LLaVA-NeXT-Interleave: Tackling Multi-image, Video, and 3D in Large Multimodal Models*. 2024. arXiv: 2407.07895 [cs.CV]. URL: https://arxiv.org/abs/2407.07895 (cit. on p. 14).

[34] Peng Wang et al. *Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution*. 2024. arXiv: 2409.12191 [cs.CV]. URL: https://arxiv.org/abs/2409.12191 (cit. on p. 14).

[35] Shuai Bai et al. *Qwen2.5-VL Technical Report*. 2025. arXiv: 2502.13923 [cs.CV]. URL: https://arxiv.org/abs/2502.13923 (cit. on p. 14).

[36] Zhe Chen et al. *Expanding Performance Boundaries of Open-Source Multimodal Models with Model, Data, and Test-Time Scaling*. 2025. arXiv: 2412.05271 [cs.CV]. URL: https://arxiv.org/abs/2412.05271 (cit. on p. 14).

[37] Zhe Chen et al. *InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks*. 2024. arXiv: 2312.14238 [cs.CV]. URL: https://arxiv.org/abs/2312.14238 (cit. on p. 14).

[38] Xiang Yue et al. *MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI*. 2024. arXiv: 2311.16502 [cs.CL]. URL: https://arxiv.org/abs/2311.16502 (cit. on p. 14).

[39] Jinguo Zhu et al. *InternVL3: Exploring Advanced Training and Test-Time Recipes for Open-Source Multimodal Models*. 2025. arXiv: 2504.10479 [cs.CV]. URL: https://arxiv.org/abs/2504.10479 (cit. on p. 14).

[40] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2024. arXiv: 2312.10997 [cs.CL]. URL: https://arxiv.org/abs/2312.10997 (cit. on pp. 14, 16).

[41] Xianzheng Ma et al. *When LLMs step into the 3D World: A Survey and Meta-Analysis of 3D Tasks via Multi-modal Large Language Models*. 2024. arXiv: 2405.10255 [cs.CV]. URL: https://arxiv.org/abs/2405.10255 (cit. on p. 15).

[42] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. *From Exploration to Mastery: Enabling LLMs to Master Tools via Self-Driven Interactions*. 2025. arXiv: 2410.08197 [cs.CL]. URL: https://arxiv.org/abs/2410.08197 (cit. on p. 15).

[43] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. 2021. arXiv: 2005.11401 [cs.CL]. URL: https://arxiv.org/abs/2005.11401 (cit. on p. 15).

[44] Liangyu Zha et al. *TableGPT: Towards Unifying Tables, Nature Language and Commands into One GPT*. 2023. arXiv: 2307.08674 [cs.AI]. URL: https://arxiv.org/abs/2307.08674 (cit. on p. 16).

[45] Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Martin Eisenschlos. *Open Domain Question Answering over Tables via Dense Retrieval*. 2021. arXiv: 2103.12011 [cs.CL]. URL: https://arxiv.org/abs/2103.12011 (cit. on p. 16).

[46] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. «TaPas: Weakly Supervised Table Parsing via Pretraining». In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. DOI: 10.18653/v1/2020.acl-main.398. URL: http://dx.doi.org/10.18653/v1/2020.acl-main.398 (cit. on p. 16).

[47] Zhiruo Wang, Zhengbao Jiang, Eric Nyberg, and Graham Neubig. *Table Retrieval May Not Necessitate Table-specific Model Design*. 2022. arXiv: 2205.09843 [cs.CL]. URL: https://arxiv.org/abs/2205.09843 (cit. on p. 16).

[48] Kihun Kim, Mintae Kim, Hokyung Lee, Seongik Park, Youngsub Han, and Byoung-Ki Jeon. «THoRR: Complex Table Retrieval and Refinement for RAG». In: *Proceedings of the Workshop Information Retrieval's Role in RAG Systems (IR-RAG 2024)*. Vol. 3784. 2024, pp. 50–55 (cit. on p. 17).

[49] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. *G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering*. 2024. arXiv: 2402.07630 [cs.LG]. URL: https://arxiv.org/abs/2402.07630 (cit. on p. 17).

[50] Weijian Xie, Xuefeng Liang, Yuhui Liu, Kaihua Ni, Hong Cheng, and Zetian Hu. *WeKnow-RAG: An Adaptive Approach for Retrieval-Augmented Generation Integrating Web Search and Knowledge Graphs*. 2024. arXiv: 2408.07611 [cs.CL]. URL: https://arxiv.org/abs/2408.07611 (cit. on p. 17).

[51] Kunal Sawarkar, Abhilasha Mangal, and Shivam Raj Solanki. «Blended RAG: Improving RAG (Retriever-Augmented Generation) Accuracy with Semantic Search and Hybrid Query-Based Retrievers». In: *2024 IEEE 7th International Conference on Multimedia Information Processing and Retrieval (MIPR)*. Vol. 24. IEEE, Aug. 2024, pp. 155–161. DOI: 10.1109/mipr62202.2024.00031. URL: http://dx.doi.org/10.1109/MIPR62202.2024.00031 (cit. on p. 18).

[52] Hamin Koo, Minseon Kim, and Sung Ju Hwang. *Optimizing Query Generation for Enhanced Document Retrieval in RAG*. 2024. arXiv: 2407.12325 [cs.IR]. URL: https://arxiv.org/abs/2407.12325 (cit. on p. 18).

[53] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. «The Power of Noise: Redefining Retrieval for RAG Systems». In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR 2024. ACM, July 2024, pp. 719–729. DOI: 10.1145/3626772.3657834. URL: http://dx.doi.org/10.1145/3626772.3657834 (cit. on p. 18).

[54] Zhengyuan Zhu, Daniel Lee, Hong Zhang, Sai Sree Harsha, Loic Feujio, Akash Maharaj, and Yunyao Li. *MuRAR: A Simple and Effective Multimodal Retrieval and Answer Refinement Framework for Multimodal Question Answering*. 2025. arXiv: 2408.08521 [cs.IR]. URL: https://arxiv.org/abs/2408.08521 (cit. on p. 18).

[55] Darren Edge et al. *From Local to Global: A Graph RAG Approach to Query-Focused Summarization*. 2025. arXiv: 2404.16130 [cs.CL]. URL: https://arxiv.org/abs/2404.16130 (cit. on p. 19).

[56] Philippe Laban, Alexander R. Fabbri, Caiming Xiong, and Chien-Sheng Wu. *Summary of a Haystack: A Challenge to Long-Context LLMs and RAG Systems*. 2024. arXiv: 2407.01370 [cs.CL]. URL: https://arxiv.org/abs/2407.01370 (cit. on p. 19).

[57] Sukmin Cho, Soyeong Jeong, Jeongyeon Seo, Taeho Hwang, and Jong C. Park. *Typos that Broke the RAG's Back: Genetic Attack on RAG Pipeline by Simulating Documents in the Wild via Low-level Perturbations*. 2024. arXiv: 2404.13948 [cs.CL]. URL: https://arxiv.org/abs/2404.13948 (cit. on p. 19).

[58] Pankaj Joshi, Aditya Gupta, Pankaj Kumar, and Manas 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, pp. 993–999. DOI: 10.1109/ICAAIC60222.2024.10574972 (cit. on p. 20).

[59] Shi Yu et al. *VisRAG: Vision-based Retrieval-augmented Generation on Multi-modality Documents*. 2025. arXiv: 2410.10594 [cs.IR]. URL: https://arxiv.org/abs/2410.10594 (cit. on p. 20).

[60] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV]. URL: https://arxiv.org/abs/1405.0312 (cit. on p. 20).

[61] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. *Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models*. 2016. arXiv: 1505.04870 [cs.CV]. URL: https://arxiv.org/abs/1505.04870 (cit. on p. 20).

[62] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. *VQA: Visual Question Answering*. 2016. arXiv: 1505.00468 [cs.CL]. URL: https://arxiv.org/abs/1505.00468 (cit. on p. 20).

[63] Ziheng Jia et al. *VQA^2: Visual Question Answering for Video Quality Assessment*. 2024. arXiv: 2411.03795 [cs.CV]. URL: https://arxiv.org/abs/2411.03795 (cit. on p. 20).

[64] Nitesh Methani, Pritha Ganguly, Mitesh M. Khapra, and Pratyush Kumar. *PlotQA: Reasoning over Scientific Plots*. 2020. arXiv: 1909.00997 [cs.CV]. URL: https://arxiv.org/abs/1909.00997 (cit. on p. 21).

[65] Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Akos Kadar, Adam Trischler, and Yoshua Bengio. *FigureQA: An Annotated Figure Dataset for Visual Reasoning*. 2018. arXiv: 1710.07300 [cs.CV]. URL: https://arxiv.org/abs/1710.07300 (cit. on p. 21).

[66] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. *Towards VQA Models That Can Read*. 2019. arXiv: 1904.08920 [cs.CL]. URL: https://arxiv.org/abs/1904.08920 (cit. on p. 21).

[67] Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusiñol, Ernest Valveny, C. V. Jawahar, and Dimosthenis Karatzas. *Scene Text Visual Question Answering*. 2019. arXiv: 1905.13648 [cs.CV]. URL: https://arxiv.org/abs/1905.13648 (cit. on p. 21).

[68] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. *ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning*. 2022. arXiv: 2203.10244 [cs.CL]. URL: https://arxiv.org/abs/2203.10244 (cit. on p. 21).

[69] Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. *DocVQA: A Dataset for VQA on Document Images*. 2021. arXiv: 2007.00398 [cs.CV]. URL: https://arxiv.org/abs/2007.00398 (cit. on p. 21).

[70] Minesh Mathew, Viraj Bagal, Rubèn Pérez Tito, Dimosthenis Karatzas, Ernest Valveny, and C. V Jawahar. *InfographicVQA*. 2021. arXiv: 2104.12756 [cs.CV]. URL: https://arxiv.org/abs/2104.12756 (cit. on p. 21).

[71] Rubèn Tito, Dimosthenis Karatzas, and Ernest Valveny. *Hierarchical multimodal transformers for Multi-Page DocVQA*. 2023. arXiv: 2212.05935 [cs.CV]. URL: https://arxiv.org/abs/2212.05935 (cit. on p. 21).

[72] Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. *MultiModalQA: Complex Question Answering over Text, Tables and Images*. 2021. arXiv: 2104.06039 [cs.CL]. URL: https://arxiv.org/abs/2104.06039 (cit. on p. 22).

[73] Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. *WebQA: Multihop and Multimodal QA*. 2022. arXiv: 2109.00590 [cs.CL]. URL: https://arxiv.org/abs/2109.00590 (cit. on p. 22).

[74] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. *SEED-Bench: Benchmarking Multimodal LLMs with Generative Comprehension*. 2023. arXiv: 2307.16125 [cs.CL]. URL: https://arxiv.org/abs/2307.16125 (cit. on p. 22).

[75] Yuan Liu et al. *MMBench: Is Your Multi-modal Model an All-around Player?* 2024. arXiv: 2307.06281 [cs.CV]. URL: https://arxiv.org/abs/2307.06281 (cit. on p. 22).

[76] Shraman Pramanick, Rama Chellappa, and Subhashini Venugopalan. *SPIQA: A Dataset for Multimodal Question Answering on Scientific Papers*. 2025. arXiv: 2407.09413 [cs.CL]. URL: https://arxiv.org/abs/2407.09413 (cit. on p. 22).

[77] Mehran Kazemi et al. *ReMI: A Dataset for Reasoning with Multiple Images*. 2024. arXiv: 2406.09175 [cs.CV]. URL: https://arxiv.org/abs/2406.09175 (cit. on p. 23).

[78] Danae Sánchez Villegas, Ingo Ziegler, and Desmond Elliott. *ImageChain: Advancing Sequential Image-to-Text Reasoning in Multimodal Large Language Models*. 2025. arXiv: 2502.19409 [cs.CV]. URL: https://arxiv.org/abs/2502.19409 (cit. on p. 23).

[79] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. «Bleu: a method for automatic evaluation of machine translation». In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318 (cit. on p. 23).

[80] Chin-Yew Lin. «Rouge: A package for automatic evaluation of summaries». In: *Text summarization branches out*. 2004, pp. 74–81 (cit. on p. 23).

[81] ExplodingGradients. *Ragas: Supercharge Your LLM Application Evaluations*. `https://github.com/explodinggradients/ragas`. 2024 (cit. on p. 35).

[82] langchain-ai. *Langchain - Build context-aware reasoning applications*. `https://github.com/langchain-ai/langchain`. 2025 (cit. on p. 36).