UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Dottorato di ricerca in
Information and communication technologies (ICT)

Ciclo XXXVIII

Improving Retrieval Augmented Generation Pipelines for Question Answering over Documents

Candidato:
Francesco Maria Granata

Relatore (Tutor):
Prof. Baraldi Lorenzo

Correlatore (Co-Tutor):
Dott. Costa Davide

Coordinatore:
Prof. Rovati Luigi

Review committee composed of:
Filippo Furfaro, Università della Calabria
Samuele Poppi, MBZUAI

Contents

Chapter 1

Introduction

Natural Language Processing (NLP) has undergone a remarkable transformation over the past decade, evolving from traditional extractive approaches to sophisticated generative systems capable of synthesizing comprehensive responses from multiple sources. In this context, the Retrieval-Augmented Generation (RAG) paradigm has emerged as a pivotal solution to address the fundamental limitations of Large Language Models (LLMs), such as knowledge cutoff and the phenomenon of hallucination. This thesis explores innovative methodologies to improve RAG pipelines for Question Answering (QA) over documents.

1.1 Overview

RAG systems represent a natural convergence of generative language models and dense retrieval methods. This modular architecture allows for grounding responses in external, verifiable evidence, enabling knowledge updates without the need for extensive model retraining. However, the effectiveness of these pipelines depends on the seamless integration of several components: from document ingestion and chunking to semantic retrieval and final response synthesis. While the individual components of the RAG pipeline—from retrieval to generation—are often evaluated independently, this thesis posits that true optimization requires a unified perspective. The proposed methodologies are deeply interrelated: robust multilingual benchmarking for Information Retrieval provides the foundation for selecting base models, which are then specialized through the SAGE domain

adaptation framework 3. Similarly multilingual benchmarking is crucial to select optimal Chat Models for Question Answering tasks, which may require further fine-tuning 4. However, even highly adapted models depend on optimal, domain-aware chunking strategies to balance context coherence with retrieval precision. Ultimately, these upstream optimizations in retrieval and segmentation culminate in the system's ability to generate faithful answers and provide robust source attribution, effectively closing the loop on a reliable, end-to-end intelligent document processing pipeline 5.

1.2 Motivation and Objectives

1.2.1 Challenges in Intelligent Document Processing

Integrating advanced QA systems into Intelligent Document Processing (IDP) platforms faces several technical challenges:

- **Multilingual and Domain Gaps:** Many models are primarily optimized for English, showing significant performance degradation in other languages like Italian or in specialized domains such as finance, law, and medicine.
- **Evaluation of Generative Systems:** The rapid proliferation of LLMs necessitates systematic benchmarking frameworks that evaluate not only fluency but also groundedness (faithfulness to the context) and relevance.
- **Context Management:** Models often struggle with long-range dependencies and the "Lost in the Middle" phenomenon, where critical information in the center of long inputs is overlooked.
- **Complex Data Structures:** Real-world documents frequently organize information in tables, requiring specialized reasoning that differs from sequential text processing.

1.2.2 Research Objectives

The primary goal of this research is the end-to-end optimization of RAG pipelines. Specifically, the thesis aims to:

- **Systematic Benchmarking:** Rigorously evaluate the cross-lingual and cross-domain capabilities of both state-of-the-art embedding models and chat models.

- **Evaluation Methodology:** Analyze and categorize evaluation frameworks for RAG, addressing the limitations of traditional metrics in assessing generative accuracy.
- **Dynamic Adaptation:** Develop scalable frameworks for adapting embedding models to specialized domains in low-resource settings.
- **Structural Optimization:** Investigate optimal document segmentation (chunking) strategies in relation to the capabilities of long-context LLMs.
- **Reliability and Verifiability:** Enhance source attribution mechanisms to ensure that generated responses are transparent and grounded in the retrieved evidence.

1.3 Thesis Contributions

1.3.1 Scientific Contributions

This thesis situates itself at the intersection of several critical debates within the contemporary Natural Language Processing landscape, providing empirical grounding to ongoing discussions on retrieval-generation integration, domain specialization, and Trustworthy AI. At its foundation, this work addresses the community's pressing need for rigorous, reproducible baselines through systematic benchmarking. We provide a comprehensive study evaluating 12 embedding models across English and Italian, highlighting the effectiveness of multilingual pre-training over language-specific models. Concurrently, our systematic evaluation of proprietary and open-weight LLMs identifies the "groundedness gap" between response relevance and factual fidelity. To accurately measure these phenomena, the thesis presents a critical analysis and taxonomy of evaluation methodologies at both the Information Retrieval (e.g., NDCG, MAP) and Question Answering levels. This includes a review of syntactic, semantic (e.g., BEM), and reference-free LLM-based metrics (e.g., RAGAS, TruLens). Building on these foundational insights, the research contributes directly to the discourse on efficient model specialization through dynamic adaptation. Rather than relying on massive, monolithic black-box models, we introduce the SAGE framework, which integrates synthetic data generation and cached contrastive learning for unsupervised domain adaptation. Furthermore, we optimize the handling of complex data structures by investigating table representations and response paradigms, demonstrating that Direct QA often outperforms Semantic Parsing for well-structured

data. Finally, while recent trends in foundation models emphasize scaling context windows to process entire documents intrinsically, our chunking analysis provides empirical evidence showing that structural chunking remains superior to whole-document ingestion even for long-context models, particularly in semantically dense domains. To further ensure Trustworthy AI, this research tackles the context-attribution problem by developing semi-supervised cross-encoders trained on synthetically generated data. This methodology offers a highly efficient alternative to LLM-based attribution, proving that lightweight models can reliably link generated answers to specific retrieved passages without prohibitive computational costs or the need for extensive manual annotation . Ultimately, by integrating these systematic evaluations and structural optimizations, this thesis advocates for a modular, resource-efficient approach to Intelligent Document Processing.

1.3.2 Industrial Contributions

Beyond its scientific advancements, this research has directly influenced the Altilia platform by translating theoretical insights into tangible industrial applications. A primary contribution is the establishment of evidence-based model selection guidelines, which provide robust strategies for choosing optimal model combinations tailored to multilingual and specialized enterprise environments. Building on this foundation, the integration of the Embedding Models Domain Adaptation framework allows the platform to automatically adapt embedding models to newer domains employing synthetic data generation without relying on expensive manual annotation. To ensure these systems evolve and learn from real-world usage, we designed the RAG-Dataset format. This multi-level annotation schema uniquely supports both chunk and answer feedback, creating a structured mechanism for the continuous improvement of production systems. Moreover, the integration of our production-ready source attribution methodology enables the platform to provide verifiable and traceable responses. This capability fulfills stringent regulatory compliance requirements in high-stakes sectors, such as finance and law, while simultaneously bypassing the need for expensive proprietary APIs . Finally, these structural and methodological improvements collectively drive significant operational efficiency. By demonstrating how specialized open-weight models can be effectively deployed, this work provides a blueprint for reducing inference costs and mitigating privacy risks while consistently maintaining high performance in commercial settings.

1.4 Thesis Organization

The remainder of this thesis is organized as follows: Chapter 2 reviews the technological background and evaluation challenges; Chapter 3 focuses on Information Retrieval and the SAGE framework; Chapter 4 analyzes Question Answering, including chat model benchmarks and tabular data; Chapter 5 presents RAG optimizations, including chunking and source attribution; finally, Chapter 6 provides concluding remarks and future perspectives.

1.5 List of Publications

- **Evaluating retrieval-augmented generation for question answering with large language models**, CEUR Workshop Proceedings, 2024
- **Leveraging Large Language Models for Flexible and Robust Table-to-Text Generation**, DEXA, 2024
- **A Comprehensive Evaluation of Embedding Models and LLMs for IR and QA Across English and Italian**, Big Data and Cognitive Computing 9 (5), 2025
- **Improving Context-Attribution with Semi-Supervised Cross-Encoders**, ECAI, 2025

Chapter 2

Background and Technological Context

This chapter provides a comprehensive overview of the technological foundations underlying the research presented in this thesis. We examine the evolution of Natural Language Processing (NLP) from traditional approaches to modern transformer-based architectures, explore the transformation of Information Retrieval through dense embedding techniques, and discuss how these advances have reshaped Question Answering systems within Intelligent Document Processing platforms. The chapter concludes with an analysis of the challenges associated with evaluating conversational and generative AI systems. Understanding these foundational concepts is essential for appreciating the contributions made in subsequent chapters regarding Retrieval-Augmented Generation (RAG) systems and question answering over documents.

2.1 Evolution of NLP and Language Models

Natural Language Processing has undergone a remarkable transformation over the past decade, driven by advances in deep learning and the availability of large-scale computational resources. This evolution has fundamentally changed how machines understand and generate human language, paving the way for sophisticated applications in information retrieval and question answering [16].

Early neural approaches to NLP relied heavily on sequence-to-sequence

(Seq2Seq) architectures employing recurrent neural networks (RNNs), particularly Long Short-Term Memory (LSTM) networks. While successful in tasks like translation, they suffered from fundamental limitations in parallelization and handling long-range dependencies [80]. The introduction of the Transformer architecture by Vaswani et al. [84] marked a paradigm shift by abandoning recurrence in favor of the *self-attention mechanism*. This mechanism computes a representation of the sequence by relating different positions of a single sequence. Formally, given queries (Q) and keys (K) of dimension d_k, and values (V) of dimension d_v, the attention function is computed as the weighted sum of the values. The weights are determined by the compatibility function of the query with the corresponding key, defined as Scaled Dot-Product Attention:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

The scaling factor $\frac{1}{\sqrt{d_k}}$ is crucial to counteract the effect of large dot products pushing the softmax function into regions with vanishing gradients. To capture relationships from different representation subspaces, the Transformer employs *Multi-Head Attention*, which linearly projects the queries, keys, and values h times with different, learned linear projections:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{2.2}$$

where each head is computed as $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$. These properties—scalability and parallelization—have made the Transformer the de-facto standard for sequence-to-sequence tasks.

The Transformer architecture established two primary paradigms in language modeling. *Encoder-decoder* architectures, exemplified by the original Transformer and T5 [69], process an input sequence to create contextualized representations before generating an output sequence, a design well-suited for translation and summarization. Conversely, *Decoder-only* architectures utilize masked self-attention—where the attention mask ensures that position i can only attend to positions $j < i$—to process text autoregressively. The simplicity of the decoder-only design has since become the dominant paradigm for modern Large Language Models (LLMs).

Two foundational lineages have shaped this landscape: BERT and the GPT series. BERT (Bidirectional Encoder Representations from Transformers) [16] revolutionized NLP by employing bidirectional pre-training via masked language modeling (MLM). Unlike unidirectional models, BERT considers context from

both directions simultaneously, making it highly effective for understanding tasks such as classification and named entity recognition. Parallel to this, the GPT (Generative Pre-trained Transformer) series [67, 7] demonstrated the efficacy of the decoder-only architecture at scale. The evolution from GPT through GPT-4 proved that increasing parameter count and training data yields emergent properties, such as few-shot learning. GPT-3's 175 billion parameters established prompting as a primary method for task specification, reducing the need for extensive fine-tuning.

The current era is defined by the dichotomy between closed-source proprietary models and open-weight alternatives. While proprietary models like GPT-4 initially led in performance, the release of LLaMA [82] marked a turning point, demonstrating that smaller, efficient models trained on vast datasets could rival larger counterparts. This sparked a surge in open-weight development, led by the LLaMA and Mistral [32] families. These models are typically "instruction-tuned" to align with human intent. Furthermore, the ecosystem has expanded to include specialized open models like Mistral-Nemo and Google's Gemma [25]. These recent additions specifically target the performance gap between high-resource and lower-resource languages, responding to the growing demand for efficient multilingual solutions [60]. This democratization of large language model technology has enabled broader adoption in research and industry, particularly for applications requiring data privacy or local inference.

2.2 Dense Retrieval and Embedding Models

The advances in transformer-based language models have profoundly transformed information retrieval, giving rise to dense retrieval methods that represent a fundamental departure from traditional keyword-based approaches. Traditional information retrieval systems relied on lexical matching techniques, with BM25 [74] being the most prominent example. These methods match queries to documents based on term frequency and inverse document frequency, treating words as discrete symbols without semantic understanding. While effective for many use cases, lexical approaches struggle with vocabulary mismatch—the phenomenon where queries and relevant documents use different words to express the same concepts.

Dense retrieval methods address this limitation by representing both queries and documents as dense vectors in a shared semantic space. Rather than matching exact terms, these systems compute similarity in the embedding space, enabling retrieval based on meaning rather than surface form. This approach excels at capturing complex semantic relationships, handling synonymy and polysemy, and

retrieving relevant documents even when query and document terminology differ significantly. Two prominent architectural paradigms have emerged for neural retrieval. Bi-encoders encode queries and documents independently using the same or separate encoders, producing fixed-dimensional vector representations; the relevance score is typically computed as the cosine similarity or dot product between query and document vectors. Formally, let $f_\Theta : \mathcal{X} \to \mathbb{R}^d$ denote an encoder that maps text sequences to d-dimensional embeddings. Given a query q and a document d, **cosine similarity** computes the normalized inner product:

$$\text{sim}_{\cos}(q, d) = \frac{f_\Theta(q) \cdot f_\Theta(d)}{\|f_\Theta(q)\| \cdot \|f_\Theta(d)\|} \tag{2.3}$$

When embeddings are L2-normalized (i.e., $\|f_\Theta(x)\| = 1$), cosine similarity reduces to the **dot product**:

$$\text{sim}_{\text{dot}}(q, d) = f_\Theta(q)^\top \cdot f_\Theta(d) \tag{2.4}$$

This formulation enables retrieval through approximate nearest neighbor search algorithms such as HNSW or IVF, since document embeddings can be pre-computed and indexed offline, enabling efficient search over millions of documents. Pioneering bi-encoder methods include Dense Passage Retrieval (DPR) [34], ColBERT [35], and ANCE [94]. Cross-encoders, by contrast, process the query and document jointly, allowing full attention between all tokens. While this approach typically achieves higher accuracy by modeling fine-grained interactions, it requires computing representations for each query-document pair at query time, making it computationally prohibitive for large-scale retrieval. Cross-encoders are therefore commonly used as re-rankers, refining the results of an initial bi-encoder retrieval stage.

The field has seen rapid development of specialized embedding models designed explicitly for retrieval tasks. The E5 family [86] introduced a simple yet effective approach using prefix-based input formatting to distinguish between queries and passages. The GTE (General Text Embeddings) models [45] achieved strong performance through contrastive learning on large-scale paired data, while BGE (BAAI General Embeddings) [92] demonstrated robust multilingual performance by leveraging advanced pre-training methods such as RetroMAE. For multilingual applications, models like multilingual-E5 have proven particularly valuable, offering consistent performance across languages through training on diverse multilingual corpora. Language-specific models have also emerged to

address unique linguistic characteristics; for instance, BERTino provides an Italian-optimized alternative to general multilingual models.

Modern embedding models are typically trained using contrastive learning objectives, which encourage the model to produce similar representations for semantically related text pairs while pushing apart unrelated pairs. The Multiple Negative Ranking (MNR) loss has become a standard approach, leveraging in-batch negatives to efficiently train bi-encoders without explicit hard negative mining. This training paradigm has enabled the development of domain-adapted embedding models through fine-tuning on domain-specific data, a capability that proves essential for specialized applications in finance, legal, or technical domains.

2.3 Question Answering in Intelligent Document Processing

Question Answering represents one of the core capabilities of Intelligent Document Processing platforms, enabling users to extract information from large collections of complex documents using natural language queries. This functionality significantly reduces the time required to analyze documents and democratizes access to information that would otherwise require extensive manual review. QA systems have undergone a fundamental transformation over the past years, evolving from extractive approaches that identify answer spans within source documents to generative systems capable of synthesizing comprehensive responses from multiple sources.

Traditional extractive QA systems identify specific spans of text that answer a given question. Models like BERT fine-tuned on SQuAD [71] achieved impressive performance on this task by learning to predict the start and end positions of answer spans within passages. This approach works well when the exact answer text appears verbatim in the source document, but it faces limitations when answers require synthesis across multiple passages, when the information must be reformulated, or when the exact answer text is not present in the source documents. Large language models have enabled generative QA, where answers are generated rather than extracted. This approach allows for more natural and comprehensive responses that can synthesize information from multiple sources, explain reasoning, and adapt the response format to the question type. The integration of retrieval with generation, known as Retrieval-Augmented Generation (RAG), has emerged as a powerful paradigm for knowledge-intensive QA tasks [42], combining the parametric knowledge of language models with the ability to access and

ground responses in external documents. The RAG paradigm represents a natural convergence of the technological advances discussed in the preceding sections: transformer-based language models provide the generative capabilities, while dense retrieval methods enable efficient access to relevant knowledge. This modular architecture offers several compelling advantages for document-centric QA applications. First, it enables knowledge updates without model retraining, as the document collection can be modified independently of the generation component. Second, retrieved passages provide explicit evidence for generated answers, enhancing interpretability and enabling users to verify the sources underlying system responses. Third, it reduces the computational and data requirements compared to training or fine-tuning models to internalize domain-specific knowledge. These properties make RAG particularly well-suited for Intelligent Document Processing platforms, where document collections evolve continuously and where transparency and auditability are often essential requirements. Modern conversational AI systems extend beyond single-turn QA to maintain context across multiple interactions. These systems must manage dialogue state, handle follow-up questions that depend on conversational history, resolve coreferences, and provide coherent responses that account for the entire conversation context. The integration of RAG pipelines with conversational interfaces has enabled more accurate and contextually aware interactions, reducing hallucinations and improving factual grounding [78]. A particular challenge in conversational QA is the "Lost in the Middle" phenomenon identified by Liu et al. [48], where models struggle to maintain attention across long input contexts. This limitation is particularly relevant for QA systems processing extensive documents or integrating information from multiple retrieved passages, motivating research into more effective context management strategies. Despite significant advances in AI-powered Question Answering, human oversight remains essential for ensuring accuracy, handling edge cases, and maintaining quality in production systems. The human-in-the-loop paradigm integrates human judgment at strategic points in automated workflows. Systems can intelligently select documents or predictions for human review, focusing expert attention where it is most needed; by prioritizing uncertain or potentially erroneous predictions through active learning approaches, this strategy maximizes the value of human review while minimizing annotation burden. Human reviewers validate AI outputs and provide corrections that feed back into model improvement, a loop that proves crucial for domain adaptation and continuous improvement of Intelligent Document Processing systems. Complex or unusual documents that fall outside the training distribution can be routed to human experts, ensuring robust handling of edge cases while maintaining high throughput for routine documents.

2.4 Challenges in Evaluating Conversational and Generative Systems

Evaluating the quality of outputs from conversational and generative AI systems presents unique challenges that traditional NLP evaluation methods are ill-equipped to address. Unlike classification tasks with clear ground truth labels, generative tasks admit multiple valid outputs, and quality assessment requires considering dimensions such as factual accuracy, relevance, coherence, and faithfulness to source material.

Traditional evaluation metrics for text generation were developed primarily for tasks like machine translation and summarization, where there is typically a clear reference output against which generated text can be compared. BLEU (Bilingual Evaluation Understudy), originally designed for machine translation [62], measures n-gram overlap between generated and reference texts. While useful for assessing surface-level similarity, BLEU fails to capture semantic equivalence when different phrasings convey the same meaning. This limitation is particularly problematic for QA systems, where correct answers may be expressed in many equally valid ways. ROUGE (Recall-Oriented Understudy for Gisting Evaluation), developed for summarization evaluation [47], assesses recall of n-grams, longest common subsequences, or skip-bigrams. Like BLEU, ROUGE is limited to lexical overlap and cannot adequately evaluate semantic correctness of generated responses. BERTScore represents a more recent approach that leverages BERT embeddings to compute semantic similarity between generated and reference texts [101]. BERTScore addresses some limitations of lexical metrics by capturing semantic similarity, but still requires reference texts for comparison and may not fully capture the nuances of factual accuracy or response appropriateness.

Human evaluation, while providing the most reliable assessment of response quality, is expensive, time-consuming, and difficult to scale. This has motivated the development of automatic evaluation approaches that can approximate human judgment. Large language models can be employed as evaluators, assessing generated responses for various quality dimensions such as relevance, coherence, and factual accuracy. Frameworks like RAGAS [19] and TruLens implement LLM-based evaluation metrics including context relevance (whether retrieved passages are relevant for answering the question), groundedness (whether the generated answer is faithful to the retrieved passages), and answer relevance (whether the response appropriately addresses the query). These metrics offer scalable evalu-

ation without requiring reference answers, though their reliability varies across domains and question types. Developing metrics that can assess response quality without reference answers remains an active research area. Current reference-free approaches show poor correlation with human judgment, highlighting that evaluating answers without ground truth remains a challenging problem for large language models [59]. The difficulty lies in distinguishing between responses that are fluent but factually incorrect and those that are correct but stylistically different from expected answers.

A fundamental challenge in evaluating and training conversational AI systems is the scarcity of high-quality annotated data, particularly for domain-specific applications. Creating gold-standard QA pairs requires domain expertise and significant human effort; for specialized domains such as finance, legal, or ESG reporting, annotation requires experts who understand both the domain and the nuances of question-answer evaluation. This creates significant bottlenecks in developing and evaluating domain-specific systems. To address data scarcity, researchers have explored using LLMs to generate synthetic training data. This approach leverages large language models to create question-answer pairs from document chunks, eliminating the need for manual annotation while preserving domain-specific context. However, the quality of synthetic data depends heavily on the generation process, and evaluation benchmarks that reflect real-world domain distributions remain difficult to construct.

Chapter 3

Information Retrieval for RAG Systems

3.1 State of the Art

Information Retrieval (IR) has undergone a profound transformation over the past decade, evolving from traditional lexical matching approaches to sophisticated neural methods that capture semantic relationships between queries and documents. This evolution is particularly significant in the context of Retrieval-Augmented Generation (RAG) systems, where the quality of retrieved information directly impacts the accuracy and reliability of generated responses. Understanding the current landscape of IR techniques is therefore essential for designing effective RAG pipelines. This section provides a comprehensive overview organized into five main areas: traditional lexical retrieval, dense neural retrieval, hybrid approaches, reranking models, and benchmarking frameworks.

3.1.1 Traditional Retrieval: BM25 and Lexical Methods

Traditional IR methods are based on lexical matching between query terms and document terms, with BM25 representing the most widely adopted approach in this category [74]. BM25 extends the classic TF-IDF weighting scheme [75] by incorporating document length normalization and term frequency saturation, making it particularly effective for ad-hoc retrieval tasks. The BM25 scoring

function for a document D given a query $Q = \{q_1, q_2, \ldots, q_n\}$ is formally defined as:

$$\text{BM25}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)} \tag{3.1}$$

where:

- $f(q_i, D)$ is the term frequency of query term q_i in document D
- $|D|$ is the document length (number of terms)
- avgdl is the average document length across the corpus
- k_1 controls term frequency saturation (typically $k_1 \in [1.2, 2.0]$)
- b controls document length normalization (typically $b = 0.75$)

The **Inverse Document Frequency (IDF)** component penalizes terms that appear in many documents:

$$\text{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1 \tag{3.2}$$

where N is the total number of documents in the corpus and $n(q_i)$ is the number of documents containing term q_i. The additive constant ensures non-negative IDF values even for very frequent terms. The algorithm computes relevance scores based on three fundamental components: term frequency in documents, inverse document frequency across the corpus, and document length normalization parameters that prevent bias toward longer documents. Despite the advent of neural methods, BM25 remains remarkably competitive and continues to serve as a strong baseline in many retrieval scenarios [81]. Its advantages include computational efficiency, interpretability, and the ability to perform exact keyword matching, which in some specialized domains is a key feature in retrieving useful documents. Furthermore, BM25 does not require training data and can be deployed immediately in new corpora, which makes it particularly valuable in scenarios where labeled data are scarce or unavailable. Other notable lexical methods include query likelihood models from the language modeling approach to IR [63] and divergence from randomness models [3]. Although all of these methods have been largely superseded by neural approaches in terms of absolute performance, they remain important components in production systems due to their efficiency, reliability, and the guarantees they provide for exact term matching.

3.1.2 Dense Retrieval and Neural Representations

The emergence of transformer-based architectures [84], particularly BERT [16] and its variants, has revolutionized Information Retrieval by enabling semantic representations that capture meaning beyond lexical overlap. Dense retrieval methods represent both queries and documents as continuous vectors in a shared embedding space, where semantic similarity is computed through vector operations such as dot product or cosine similarity. Dense Passage Retrieval (DPR) [34] marked a significant milestone by demonstrating that dense representations trained on question-passage pairs could substantially outperform BM25 on open-domain question answering tasks. DPR employs a bi-encoder architecture where separate BERT encoders process queries and passages independently, enabling efficient retrieval through approximate nearest neighbor search over pre-computed passage embeddings.

To overcome the bottleneck of compressing documents into a single vector, Late Interaction models have emerged as a significant architectural innovation. ColBERT [35] introduced a paradigm that preserves token-level representations until the final matching stage, computing similarity by matching all query token embeddings against all document token embeddings and aggregating the maximum similarities. This approach provides a balance between the expressiveness of cross-encoders and the efficiency of bi-encoders. Subsequent iterations like ColBERTv2 [77] addressed the storage requirements through residual compression, reducing index size by 6-10$\times$ while achieving state-of-the-art zero-shot performance on BEIR through denoised supervision. The PLAID engine [76] further improved efficiency by treating documents as bags of centroids for initial filtering, achieving substantial speedups. More recent work on XTR [41] rethinks token retrieval entirely, making the scoring stage orders of magnitude cheaper while advancing state-of-the-art performance without distillation.

Parallel to dense and late-interaction models, Learned Sparse Representations offer another neural direction that retains the interpretability of lexical methods. SPLADE [21, 20] learns sparse vocabulary-sized vectors by passing BERT output through the MLM head with log-saturation activation, enabling term expansion beyond the original document vocabulary. Where BM25 relies purely on term frequency heuristics, SPLADE learns contextual importance weights and discovers semantic relationships. Efficient SPLADE variants achieve latency comparable to BM25 through L1 regularization while substantially outperforming both pure dense retrievers and BM25 on BEIR benchmarks. Other learned sparse approaches include DeepImpact for learned term weighting compatible with standard in-

verted indexes, and document expansion techniques like docTTTTTquery [58] that generate potential queries a document might answer, significantly improving effectiveness.

The field has also witnessed substantial innovation in developing specialized embedding models optimized for retrieval tasks. Sentence-BERT [73] pioneered the use of siamese network architectures for generating semantically meaningful sentence embeddings. ANCE [94] addressed the challenge of selecting informative negative samples during training by using an asynchronously updated index to retrieve hard negatives. The E5 family of models [86] and BGE (BAAI General Embeddings) [92] demonstrated that text embeddings trained through weakly-supervised contrastive pre-training on diverse text pairs could achieve strong performance across multiple languages and retrieval tasks. For language-specific applications, specialized models have emerged to address unique linguistic characteristics, such as BERTino [54] for Italian text. Two prominent training paradigms have emerged for these embedding models: bi-encoder and cross-encoder approaches [73, 31]. Formally, let $f_\Theta : \{x_1, \ldots, x_n\} \to \mathbf{z} \in \mathbb{R}^d$ denote a bi-encoder model where f is the network architecture, Θ represents the trainable parameters, and x is the input token sequence. The **Multiple Negative Ranking (MNR) loss** leverages in-batch negatives for efficient contrastive training. Given a mini-batch $\mathcal{B} = \{(q_i, c_i^+)\}_{i=1}^{|\mathcal{B}|}$, where q_i is a query and c_i^+ is its corresponding positive passage:

$$\mathcal{L}_{\text{MNR}} = -\frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \log \frac{\exp\left(\text{sim}(f(q_i), f(c_i^+))/\tau\right)}{\sum_{j=1}^{|\mathcal{B}|} \exp\left(\text{sim}(f(q_i), f(c_j^+))/\tau\right)} \tag{3.3}$$

where $\text{sim}(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a similarity function (typically cosine similarity), and $\tau > 0$ is a temperature hyperparameter controlling the sharpness of the softmax distribution. For each query q_i, the positive passage c_i^+ serves as the target, while all other passages c_j^+ ($j \neq i$) in the batch act as in-batch negatives. This formulation enables efficient training without explicit hard negative mining, as the batch size effectively determines the number of negative samples. Contrastive learning has become the dominant training paradigm for bi-encoders, often using Multiple Negative Ranking (MNR) loss [28] or cached contrastive losses [24] to improve training efficiency. Furthermore, synthetic data generation approaches such as GPL (Generative Pseudo Labeling) [85] now allow adapting retrieval models to specific domains without manual annotation.

3.1.3 Hybrid Retrieval: Combining Approaches

Recognizing that lexical and semantic approaches capture complementary aspects of relevance, hybrid retrieval methods have gained prominence by combining both paradigms to achieve superior performance. These approaches typically integrate BM25 or other sparse retrievers with dense neural models, leveraging the strengths of each method. Hybrid retrieval addresses several limitations of pure dense retrieval: the inability to perform exact keyword matching, sensitivity to out-of-vocabulary terms, and potential failures on queries requiring precise lexical matching. By combining sparse and dense signals, hybrid systems can handle a broader range of query types effectively.

Rank fusion techniques provide straightforward mechanisms for combining results from multiple retrieval systems. Reciprocal Rank Fusion (RRF) [14] computes a combined score for each document based on its rank in each constituent system, without requiring score normalization or calibration. The **Reciprocal Rank Fusion (RRF)** score for a document d is computed as:

$$\text{RRF}(d) = \sum_{r \in R} \frac{1}{k + \text{rank}_r(d)} \tag{3.4}$$

where R is the set of ranking lists from different retrieval systems, $\text{rank}_r(d)$ is the position of document d in ranking list r, and k is a constant (typically $k = 60$) that mitigates the impact of high rankings. However, recent research has shown that alternatives can improve performance: Convex Combination (CC) outperforms RRF in both in-domain and out-of-domain settings [8], requiring only a small training set for weight tuning. **Convex Combination** directly combines normalized scores from different retrievers:

$$\text{score}_{\text{hybrid}}(d) = \alpha \cdot \text{score}_{\text{dense}}(d) + (1 - \alpha) \cdot \text{score}_{\text{sparse}}(d) \tag{3.5}$$

where $\alpha \in [0, 1]$ is a weighting parameter that can be tuned on a validation set. This approach requires score normalization but offers finer control over the contribution of each retrieval method. Score-aware hybrid retrieval using weighted combination of dense and sparse scores demonstrates average improvements of 4-8% over RRF on BEIR benchmarks. Three-way hybrid retrieval combining BM25, dense vectors, and learned sparse representations consistently outperforms two-way combinations [12].

Query expansion techniques leveraging large language models have emerged as powerful approaches for improving retrieval without modifying the underlying

index. HyDE (Hypothetical Document Embeddings) [23] introduced a paradigm shift by using LLMs to generate hypothetical documents before embedding: rather than directly embedding a query, HyDE instructs an LLM to generate what a relevant answer document might look like, then embeds that synthetic document for similarity search. This approach achieves performance comparable to supervised dense retrievers like DPR on standard benchmarks without any labeled training data. Query2Doc [87] takes a related approach by concatenating LLM-generated pseudo-documents with the original query text, improving BM25 by 3-15% on MS MARCO and benefiting both sparse and dense retrievers. The key practical tradeoff for both methods is latency overhead, as LLM inference at query time adds significant processing time.

3.1.4 Reranking Models

Reranking has become an essential component of modern retrieval pipelines, operating as a second stage that refines the initial retrieval results to improve precision. While first-stage retrievers prioritize recall and efficiency, rerankers can afford more expensive computations to achieve higher accuracy on a smaller candidate set. The standard two-stage pattern retrieves 50-100 candidates with fast methods, then reranks to select the top 5-10 documents before generation, dramatically reducing computational cost while maintaining high answer quality. Cross-encoder rerankers represent the most established approach, employing full attention over concatenated query-document pairs to compute relevance scores. MonoBERT [56] demonstrated that cross-encoder reranking dramatically outperforms retrieval-only approaches, establishing the paradigm for neural reranking. MonoT5 [57] reformulated ranking as text generation, with the model generating “true” or “false” given a query-document pair, and relevance scores derived from token probabilities. RankT5 [104] improved on MonoT5 by directly outputting numerical relevance scores and fine-tuning with ranking-specific losses such as RankNet and LambdaRank rather than classification losses. For production deployments, the MS MARCO MiniLM models [73] offer compelling efficiency-effectiveness tradeoffs, with the 6-layer variant achieving strong performance at approximately 1800 documents per second on GPU. The BGE reranker series [92] has emerged as a widely adopted family of cross-encoder models, with bge-reranker-v2-m3 supporting over 100 languages and 8192 token context length. The latest BGE reranker v2.5 introduces layerwise inference allowing dynamic speed-accuracy tradeoffs at runtime by selecting earlier layer cutoffs for faster but slightly degraded results. Commercial offerings such as

Cohere Rerank represent the current state-of-the-art across BEIR and multilingual retrieval benchmarks. LLM-based reranking has emerged as a powerful alternative, with listwise approaches achieving particularly strong results. RankGPT [79] demonstrated that LLMs can directly generate reordered document lists, with GPT-4 surpassing supervised models across TREC-DL and BEIR benchmarks. The approach uses a sliding window strategy to handle candidate lists exceeding context limits. Three prompting paradigms have emerged for LLM reranking: pointwise scoring, pairwise ranking prompting (PRP) [65] using comparison-based sorting with $O(N \log N)$ complexity, and listwise generation of complete rankings. PRP proves remarkably efficient, with medium-sized models performing comparably to much larger models. Open-source alternatives now approach proprietary model performance at substantially lower cost. RankZephyr [64], a 7B parameter model distilled from GPT-4 listwise rankings, achieves competitive performance with BM25 retrieval on TREC benchmarks. RankLLaMA [51] applies LLaMA as pointwise rerankers, outperforming MonoT5 while remaining fully parallelizable. ListT5 [97] addresses efficiency concerns with a Fusion-in-Decoder architecture that encodes passages separately with unique identifiers, achieving improvements over RankT5 on BEIR with pointwise-comparable efficiency. FIRST [72] achieves significant latency reduction by using only first-token logits for ranking rather than generating complete permutation sequences. Multi-stage retrieval pipelines extend beyond simple two-stage architectures to achieve finer-grained relevance assessment. FunnelRAG [93] implements three stages with progressive granularity: document-level retrieval for broad coverage, passage reranking for refinement, and sentence or chunk selection for the final context, achieving substantial corpus compression with minimal answer recall degradation. The practical recommendation for most RAG systems is to start with efficient cross-encoder models such as MiniLM or bge-reranker-base for excellent cost-performance balance, scale to multilingual-capable models like bge-reranker-v2-m3 when needed, and consider LLM-based rerankers only when maximum quality justifies the computational overhead.

3.1.5 Benchmarks and Datasets

The development of comprehensive benchmarking frameworks has been crucial for advancing the field and enabling fair comparisons between retrieval methods. BEIR (Benchmarking IR) [81] established a heterogeneous benchmark for zero-shot evaluation of information retrieval models, spanning 18 diverse domains including scientific literature (SciFact), medical information (NFCorpus), argu-

ment retrieval (ArguAna), and question answering datasets (Natural Questions). By evaluating models without domain-specific fine-tuning, BEIR provides insights into generalization capabilities and has revealed significant variability in model performance across domains, highlighting the challenge of building universally effective retrieval systems. Building on BEIR, MTEB (Massive Text Embedding Benchmark) [53] expanded evaluation to eight distinct embedding tasks across multiple languages, providing a comprehensive leaderboard for comparing embedding models. The MTEB leaderboard has become an essential resource for practitioners selecting embedding models for production systems. Recent extensions such as MMTEB [18] have further expanded multilingual coverage, addressing the need for evaluation across a broader range of languages and linguistic contexts.

Standard evaluation metrics for retrieval include NDCG (Normalized Discounted Cumulative Gain), MAP (Mean Average Precision), Recall@k, and Precision@k [10]. **Normalized Discounted Cumulative Gain (NDCG)** measures ranking quality by comparing the actual ranking to an ideal ordering. The **Discounted Cumulative Gain** at position k is:

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)} \tag{3.6}$$

where rel_i is the relevance score of the item at position i. The logarithmic discount penalizes relevant documents appearing lower in the ranking. NDCG normalizes DCG by the ideal ranking:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k} \tag{3.7}$$

where $\text{IDCG@}k$ is the DCG of the ideal ranking with all relevant documents at the top. **Mean Average Precision (MAP)** averages precision across recall levels:

$$\text{MAP@}k = \frac{1}{|Q|} \sum_{q \in Q} \text{AP@}k_q = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\min(m_q, k)} \sum_{i=1}^{k} P(i) \cdot \text{rel}(i) \tag{3.8}$$

where m_q is the number of relevant documents for query q, $P(i)$ is precision at position i, and $\text{rel}(i)$ is a binary relevance indicator.

Precision@k and **Recall@k** measure retrieval accuracy and completeness:

$$\text{Precision@}k = \frac{|\text{relevant items retrieved@}k|}{k} \tag{3.9}$$

$$\text{Recall@}k = \frac{|\text{relevant items retrieved@}k|}{|\text{total relevant items}|} \tag{3.10}$$

Mean Reciprocal Rank (MRR) evaluates how early the first relevant document appears:

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i} \tag{3.11}$$

where rank_i is the position of the first relevant document for query i.
Despite significant advances, several challenges remain in information retrieval for RAG systems. Cross-lingual evaluation frameworks that assess performance across diverse language families, particularly for morphologically rich languages, remain underdeveloped [40]. The effectiveness of embedding models varies considerably across languages and domains, with performance patterns not yet systematically documented or understood, and critical questions remain about the trade-offs between model size, computational efficiency, and multilingual performance. Systematic assessment of domain adaptation capabilities when moving from general to specialized contexts presents ongoing challenges, as current systems often struggle with specialized terminology and reasoning patterns [27]. The "Lost in the Middle" phenomenon [48] has revealed that language models struggle to maintain attention across long input contexts, presenting a critical limitation for RAG systems that must integrate information from multiple retrieved passages. Addressing these challenges requires advances in both retrieval strategies and generation models, pointing toward important directions for future research in this rapidly evolving field.

3.2 Contributions

3.2.1 Multilingual Benchmarking

The development of robust benchmarking frameworks has been instrumental in advancing the field of information retrieval, enabling researchers and practitioners to systematically compare model performance and identify optimal solutions for specific applications. Prominent benchmarks such as BEIR (Benchmarking

IR) [81] and MTEB (Massive Text Embedding Benchmark) [53] have established standardized evaluation protocols that have become essential resources for the research community. However, these frameworks predominantly focus on English and a limited set of high-resource languages, creating a significant gap in our understanding of how embedding models perform across linguistic boundaries.

This limitation is particularly concerning for European languages beyond English, where the availability of evaluation resources remains limited despite substantial demand for multilingual information systems. Italian, as one of the major European languages with distinctive morphological and syntactic characteristics, represents an important test case for assessing cross-lingual transfer capabilities. To address this gap, we conducted a comprehensive multilingual benchmarking study [60] that systematically evaluates embedding models across English and Italian, providing empirical evidence of cross-lingual performance patterns and establishing new benchmarks for multilingual information retrieval systems within RAG pipelines.

Methodology Inspired by MTEB

Our evaluation framework draws methodological inspiration from the MTEB benchmark while extending its scope to address the specific requirements of cross-lingual evaluation. The MTEB framework established the practice of evaluating embedding models across multiple tasks and datasets to obtain a comprehensive picture of model capabilities. We adopted this philosophy while focusing specifically on the retrieval task, which is most directly relevant to RAG applications, and extending the evaluation to encompass both English and Italian across diverse domains.

The experimental design was structured around three complementary dimensions of analysis. First, we assessed cross-domain effectiveness by evaluating models on datasets spanning from general knowledge to highly specialized scientific and medical content. This dimension allows us to understand how well models generalize beyond their training distributions and whether certain architectural choices confer advantages in domain transfer. Second, we examined cross-language performance by comparing model behavior on parallel or comparable datasets in English and Italian, enabling direct quantification of the performance gap attributable to language differences. Third, we analyzed the impact of retrieval size on model accuracy, investigating how performance scales with the number of retrieved documents—a consideration of practical importance for RAG system design where computational costs and context window limitations must

be balanced against retrieval completeness. To evaluate these dimensions, we selected datasets covering diverse domains and linguistic contexts (summarized in Table 3.1). For the English evaluation, we employed **SQuAD** (Stanford Question

Table 3.1: Overview of datasets used for multilingual IR evaluation. The selection spans general and specialized domains across English and Italian.

Dataset	Domain	Language	Task
SQuAD-en	Open/General	English	IR
SQuAD-it	Open/General	Italian	IR
DICE	Crime News	Italian	News Retrieval
SciFact	Scientific Literature	English	Fact Checking
ArguAna	Argumentation	English	Argument Retrieval
NFCorpus	Bio-Medical	English	IR

Answering Dataset) [71] as the primary general-domain benchmark. SQuAD's widespread adoption in the NLP community and the availability of a high-quality Italian translation make it ideal for cross-lingual comparison. The dataset comprises question-answer pairs derived from Wikipedia articles, providing broad coverage of general knowledge topics. To assess performance on specialized content, we incorporated three additional English datasets from the **BEIR** benchmark collection. **SciFact** contains expert-written scientific claims paired with evidence from research abstracts, testing models' ability to retrieve relevant scientific literature for fact verification. **ArguAna** consists of argument-counterargument pairs from online debate platforms, evaluating retrieval systems' capacity to find relevant counterarguments—a task requiring nuanced semantic understanding beyond simple topical relevance. **NFCorpus** focuses on the biomedical domain with natural language queries about nutrition and health, representing one of the most challenging specialized domains due to the technical vocabulary and complex reasoning patterns involved. For the Italian evaluation on general-domain, we employed two complementary datasets. **SQuAD-it**, the Italian translation of SQuAD 1.1, enables direct comparison with English performance on equivalent content, isolating the effect of language on retrieval accuracy. As Italian specialized domain we employed **DICE** (Dataset of Italian Crime Event news) [6] corpus provides a domain-specific evaluation context with crime news articles from Italian newspapers. For the DICE evaluation, we used news titles as queries to retrieve relevant corpus documents, simulating a realistic news retrieval scenario. This

combination of general and domain-specific Italian datasets allows us to assess both baseline cross-lingual capabilities and domain adaptation in a non-English context.

Comparative Evaluation of Embedding Models

Our evaluation encompasses 12 embedding models representing the current state of the art across different design philosophies and target languages. The model selection was deliberately structured to enable multiple comparative analyses: between English-specific and multilingual models, between open-source and proprietary solutions, and between models of different sizes within the same family. Table 3.2 summarizes the models included in our evaluation.

Table 3.2: Embedding models evaluated in the multilingual benchmarking study. Models are grouped by their language scope: English-specific, multilingual, and Italian-specific.

Model	Parameters	Max Length	Language Scope
GTE-base	109M	512	English
GTE-large	335M	512	English
BGE-base-en-v1.5	109M	512	English
BGE-large-en-v1.5	335M	512	English
multilingual-E5-base	278M	512	Multilingual
multilingual-E5-large	560M	512	Multilingual
text-embedding-ada-002	N/A	8192	Multilingual
embed-multilingual-v2.0	N/A	256	Multilingual
embed-multilingual-v3.0	N/A	512	Multilingual
sentence-bert-base	109M	512	Italian
BERTino	65M	512	Italian
BERTino v2	65M	512	Italian

The English-specific models include **GTE** (General Text Embeddings) and **BGE** (BAAI General Embeddings), both available in base and large variants. These models represent strong baselines that have achieved top performance on English retrieval benchmarks and allow us to assess whether language-specific optimization provides advantages over multilingual approaches when operating within a single language.

The multilingual models constitute the largest group in our evaluation, reflecting the practical importance of cross-lingual capabilities for real-world applications. The **E5** family includes both base and large multilingual variants trained on diverse multilingual corpora using contrastive learning objectives. We also evaluated proprietary offerings from OpenAI (**text-embedding-ada-002**) and Cohere (**embed-multilingual-v2.0** and **embed-multilingual-v3.0**), which represent commercially deployed solutions with undisclosed architectural details but strong reported performance across languages.

Finally, the Italian-specific models provide a baseline for understanding whether language-specialized training offers advantages for Italian retrieval tasks. **BERTino** and its successor **BERTino v2** are DistilBERT-based models trained specifically on Italian text, while sentence-bert-base represents an Italian adaptation of the Sentence-BERT architecture. These models, while smaller in parameter count, may capture Italian-specific linguistic patterns that multilingual models might miss.

All models were evaluated in a zero-shot setting without any task-specific fine-tuning, reflecting the practical scenario where models must generalize to new domains and datasets without additional training. This evaluation paradigm provides insights into the inherent cross-domain and cross-lingual transfer capabilities of each model architecture. Following established practices in the information retrieval community and aligning with MTEB standards, we employed NDCG@10 (Normalized Discounted Cumulative Gain at rank 10) as the primary metric for assessing ranking quality. NDCG is particularly well-suited for retrieval evaluation because it accounts for both the relevance and position of retrieved items, assigning higher importance to relevant documents appearing earlier in the ranked list. To ensure the robustness of our findings and provide a more comprehensive assessment, we complemented NDCG@10 with additional metrics capturing different aspects of retrieval quality. Mean Average Precision (MAP@10) summarizes precision across different recall levels, emphasizing the importance of retrieving relevant documents early. Recall@10 measures the proportion of all relevant documents that appear in the top 10 results, assessing retrieval completeness. Precision@10 evaluates the proportion of retrieved documents that are relevant, indicating the accuracy of the retrieval system. This multi-metric approach allows us to verify that performance patterns observed with NDCG generalize across different evaluation perspectives. Our systematic evaluation reveals a nuanced landscape of embedding model performance, with effectiveness varying considerably across languages, domains, and model architectures. The zero-shot evaluation paradigm employed in our study provides insights into the inherent generalization capabilities of each

model, unconfounded by task-specific optimization. We organize our findings around three main themes: cross-domain effectiveness on English datasets, cross-language comparison between English and Italian, and the impact of retrieval size on performance. The evaluation on English datasets reveals significant performance variation across domains, highlighting the challenge of building universally effective retrieval systems. Table 3.3 presents the comprehensive nDCG@10 scores across all evaluated models and datasets.

Table 3.3: nDCG@10 scores for English and Italian datasets across different domains. Dashes indicate that the model was not evaluated on that dataset due to language incompatibility.

Model	**SQuAD-en**	**SQuAD-it**	**DICE**	**SciFact**	**ArguAna**	**NFCorpus**
GTE-base	0.87	—	—	0.74	0.56	0.37
GTE-large	0.87	—	—	0.74	0.57	0.38
bge-base-en-v1.5	0.86	—	—	0.74	0.64	0.37
bge-large-en-v1.5	0.89	—	—	0.75	0.64	0.38
multilingual-e5-base	0.90	0.85	0.56	0.69	0.51	0.32
multilingual-e5-large	0.91	0.86	0.64	0.70	0.54	0.34
ada-002 (OpenAI)	0.86	0.79	0.54	0.71	0.55	0.37
embed-multi-v2.0	0.84	0.79	0.64	0.66	0.55	0.32
embed-multi-v3.0	0.90	0.86	0.72	0.70	0.55	0.36
sentence-bert-base	—	0.52	0.22	—	—	—
BERTino	—	0.57	0.33	—	—	—
BERTino v2	—	0.64	0.40	—	—	—

Several important patterns emerge from the cross-domain analysis. First, no single model achieves universal superiority across all tasks, suggesting that the optimal model choice depends on the specific application context. The multilingual-E5-large model achieves the highest performance on general domain tasks with an nDCG@10 of 0.91 on SQuAD-en, demonstrating that multilingual training does not necessarily compromise performance on English tasks. However, the BGE models show particular strength on specialized content, achieving the best performance on ArguAna (0.64) and SciFact (0.75), indicating that English-focused training may provide advantages for domain-specific retrieval.

A particularly striking finding is the consistent degradation pattern observed as tasks become more specialized. Taking embed-multilingual-v3.0 as a repres-

entative example, performance decreases from 0.90 on general domain content (SQuAD) to 0.70 on scientific literature (SciFact), further declining to 0.55 on argument retrieval (ArguAna), and reaching its lowest point of 0.36 on medical domain tasks (NFCorpus). This gradient of approximately 60% relative performance loss from general to specialized domains represents a fundamental challenge that current pre-training approaches have not fully addressed. Similar patterns are observed across other multilingual models, with multilingual-E5-large showing comparable degradation from 0.91 (SQuAD) to 0.70 (SciFact) to 0.54 (ArguAna) to 0.34 (NFCorpus).

The GTE and BGE architectures demonstrate somewhat more robust adaptability to scientific and medical domains compared to their multilingual counterparts, maintaining stronger performance on SciFact and NFCorpus. This observation suggests that the diversity of training data and the specific pre-training objectives employed in English-focused models may confer advantages for technical content, even as multilingual models excel in general domains. The cross-lingual evaluation provides crucial insights into the effectiveness of different modeling approaches for Italian retrieval tasks. We compared multilingual models capable of operating in both languages against Italian-specific models trained exclusively on Italian text. This comparison allows us to assess whether language-specialized training provides advantages that outweigh the benefits of multilingual pre-training on larger and more diverse corpora. The results clearly demonstrate that multilingual models consistently outperform Italian-specific alternatives across both general and domain-specific datasets. On SQuAD-it, the best multilingual models (multilingual-E5-large and embed-multilingual-v3.0) achieve nDCG@10 scores of 0.86, substantially outperforming the best Italian-specific model, BERTino v2, which achieves only 0.64. This performance gap of 0.22 in absolute terms represents a 34% relative improvement, a substantial margin that underscores the effectiveness of multilingual pre-training for cross-lingual transfer. The pattern persists on the domain-specific DICE dataset, where embed-multilingual-v3.0 achieves 0.72 compared to 0.40 for BERTino v2. The more pronounced gap on this news retrieval task (0.32 absolute difference, 80% relative improvement) suggests that multilingual models may be particularly advantageous for domain-specific applications where Italian-specific training data is limited. To verify the robustness of these findings, we conducted a comprehensive multi-metric evaluation comparing performance across nDCG@10, MAP@10, Recall@10, and Precision@10. Table 3.4 presents these results for the SQuAD datasets in both languages. The multi-metric analysis confirms that model rankings remain largely stable regardless of the evaluation metric employed. For instance, multilingual-E5-large demon-

Table 3.4: Comprehensive evaluation with multiple IR metrics for SQuAD-en (English) and SQuAD-it (Italian). The consistency of rankings across metrics validates the robustness of our findings.

	SQuAD-en				SQuAD-it			
Model	nDCG	MAP	R@10	P@10	nDCG	MAP	R@10	P@10
multilingual-e5-base	0.90	0.87	0.98	0.098	0.85	0.80	0.97	0.097
multilingual-e5-large	0.91	0.88	1.00	0.100	0.86	0.83	0.95	0.095
embed-multi-v3.0	0.90	0.86	1.00	0.10	0.86	0.81	0.98	0.098
sentence-bert-base	—	—	—	—	0.52	0.45	0.72	0.072
BERTino	—	—	—	—	0.57	0.50	0.77	0.077
BERTino v2	—	—	—	—	0.64	0.58	0.85	0.085

strates consistently strong performance across all metrics for SQuAD-en (nDCG: 0.91, MAP: 0.88, Recall@10: 1.00, Precision@10: 0.10), with similarly robust results for SQuAD-it. This consistency across different evaluation perspectives strengthens our confidence that the observed advantages of multilingual models over Italian-specific alternatives represent genuine performance differences rather than artifacts of a particular metric choice. The embed-multilingual-v3.0 model emerges as particularly noteworthy, maintaining strong performance not only in general tasks but also demonstrating exceptional versatility across both SQuAD-it (0.86) and the specialized DICE dataset (0.72). This robust cross-domain performance suggests that recent architectural advances and training methodologies are successfully addressing the historical challenges of multilingual modeling, enabling single models to serve effectively across multiple languages and domains. Understanding how retrieval performance scales with the number of returned documents is essential for designing practical RAG systems, where the choice of retrieval size involves trade-offs between comprehensiveness, computational cost, and the constraints imposed by LLM context windows. We systematically analyzed this relationship using multilingual-E5-large on the DICE dataset, measuring Recall@k across a range of retrieval sizes. Table 3.5 presents the results of this analysis.

The data reveals a characteristic logarithmic growth pattern that can be decomposed into three distinct phases. In the initial rapid growth phase ($k = 1$ to $k = 20$), recall more than doubles from 0.335 to 0.680, indicating that the model's top-ranked documents capture a substantial portion of the relevant content. This

Table 3.5: Recall@k for multilingual-E5-large on the DICE dataset across different retrieval sizes.

k	**Recall@k**
1	0.335
5	0.535
10	0.611
20	0.680
50	0.767
100	0.827

phase offers the highest marginal returns per additional retrieved document. The moderate improvement phase ($k = 20$ to $k = 50$) shows continued but decelerating gains, with recall increasing by 0.087 to reach 0.767. Finally, the diminishing returns phase ($k > 50$) exhibits marginal improvements that decrease significantly, with recall reaching 0.827 at $k = 100$—only 0.060 higher than at $k = 50$.

These findings have direct implications for RAG system design. For most applications, retrieving between 10 and 20 documents provides an optimal balance between recall and computational efficiency. Going beyond 20 documents yields progressively smaller improvements while increasing both latency and the cognitive load on the generation model. However, for applications where maximum recall is critical—such as legal or compliance use cases where missing a relevant document could have serious consequences—the higher retrieval sizes may be justified despite their diminishing returns.

Results on English and Italian

The comprehensive evaluation across languages and domains reveals several significant findings that both advance our theoretical understanding and provide practical guidance for system deployment. These findings challenge some common assumptions about multilingual model performance while reinforcing others. Perhaps the most important finding from our evaluation is the demonstration that state-of-the-art multilingual embedding models achieve competitive performance across both English and Italian with minimal performance gaps in general domains. The embed-multilingual-v3.0 model maintains nDCG@10 scores of 0.90 for English and 0.86 for Italian on the SQuAD tasks, representing only a 4% relative decrease when moving from English to Italian. Similarly, multilingual-E5-large

achieves 0.91 and 0.86 for English and Italian respectively, with an equivalent 5% relative gap. These results are particularly encouraging because they suggest that recent advances in multilingual pre-training have successfully addressed many of the historical challenges that plagued earlier cross-lingual models. The ability to deploy a single model across multiple languages without substantial performance degradation simplifies system architecture and reduces the maintenance burden associated with managing separate models for each language. Furthermore, the fact that multilingual models often match or exceed the performance of English-specific models even on English tasks (e.g., multilingual-E5-large achieving 0.91 versus BGE-large-en-v1.5 at 0.89 on SQuAD-en) suggests that multilingual training may provide beneficial regularization effects that improve generalization. A counterintuitive finding from our evaluation is that larger models within the same family do not consistently outperform their smaller counterparts. The comparison between base and large variants reveals that architectural design choices and training methodology may have more impact on performance than parameter count alone. Consider the GTE model family: both GTE-base (109M parameters) and GTE-large (335M parameters) achieve identical nDCG@10 scores of 0.87 on SQuAD-en, despite the large variant having more than three times the parameters. The BGE family shows a modest advantage for the larger model, with BGE-large-en-v1.5 achieving 0.89 compared to 0.86 for BGE-base-en-v1.5, but this 3-point improvement may not justify the substantially increased computational costs for many applications. Similarly, the multilingual-E5 family shows only marginal improvements from base (0.90 on SQuAD-en) to large (0.91), despite doubling the parameter count from 278M to 560M. These observations have important practical implications. For resource-constrained deployments or applications requiring low latency, the base variants of these model families may offer the best efficiency-performance trade-off. The diminishing returns from scaling suggest that further improvements may require innovations in architecture or training methodology rather than simply increasing model size. Despite the impressive cross-lingual capabilities of modern embedding models, our evaluation reveals that performance consistency decreases substantially when moving from general to specialized domains. This domain specialization challenge represents a fundamental limitation that persists regardless of the evaluation metric, model architecture, or language under consideration.

The embed-multilingual-v3.0 model illustrates this pattern clearly in English tasks: achieving 0.90 nDCG@10 on general domain content (SQuAD), but dropping to 0.70 on scientific literature (SciFact), 0.55 on argument retrieval (ArguAna), and reaching only 0.36 on medical domain tasks (NFCorpus). This

represents a 60% relative performance loss from the best to worst performing domain—a gap that would significantly impact the reliability of RAG systems deployed in specialized contexts.

The pattern persists in Italian, though with fewer domain-specific datasets available for comparison. The embed-multilingual-v3.0 model achieves 0.86 nDCG@10 on the general domain (SQuAD-it) but drops to 0.72 on the specialized news domain (DICE). Language-specific models like BERTino show even more pronounced degradation, with performance dropping from 0.64 on SQuAD-it to 0.40 on DICE—a 37% relative decrease that underscores the importance of domain adaptation for non-English applications.

These findings suggest that current pre-training approaches, despite their success in general domains, do not sufficiently capture the specialized vocabulary, reasoning patterns, and domain-specific relationships required for technical content. Addressing this limitation may require domain-adaptive pre-training, specialized fine-tuning, or hybrid retrieval approaches that combine neural and lexical methods to ensure coverage of domain-specific terminology.

Optimal Model Selection for Platform

The empirical findings from our multilingual benchmarking study provide actionable guidance for model selection in production RAG systems. Rather than seeking a single optimal model, our results suggest that the best choice depends on the specific requirements of the application, including the languages to be supported, the domains to be covered, and the computational resources available. We synthesize our findings into practical recommendations for different deployment scenarios. For applications requiring consistent performance across English and Italian in general domains, embed-multilingual-v3.0 emerges as the optimal choice based on our evaluation. This model offers the best balance of cross-lingual consistency (0.90/0.86 nDCG@10 for English/Italian on SQuAD) and robust performance on domain-specific content (0.72 on DICE). The multilingual-E5-large model represents a strong open-weight alternative with comparable performance characteristics, offering the advantage of full model transparency and the possibility of fine-tuning for specific applications. The proprietary text-embedding-ada-002 from OpenAI, while widely used, shows somewhat lower performance in our evaluation (0.86/0.79 for English/Italian SQuAD), suggesting that the newer Cohere embeddings may offer advantages for multilingual applications. However, ada-002's substantially longer context window (8192 tokens versus 512 for most alternatives) may be advantageous for applications requiring processing of longer

documents. When deployment is primarily English-focused with requirements for strong performance on specialized domain content, the BGE model family (particularly bge-large-en-v1.5) provides superior performance on scientific and argumentative content. The BGE-large model achieves 0.75 on SciFact and 0.64 on ArguAna, outperforming all multilingual alternatives on these specialized tasks. This advantage comes at the cost of cross-lingual capability, making BGE models appropriate only when Italian support is not required. For applications requiring both English specialization and some multilingual capability, a hybrid approach may be optimal: using BGE models for English-specific specialized content while maintaining multilingual models for general content and other languages. For scenarios with computational constraints—such as edge deployment, high-throughput applications, or cost-sensitive cloud deployments—multilingual-E5-base offers an excellent efficiency-performance trade-off. With 278M parameters, it achieves 0.90 nDCG@10 on English SQuAD, matching the performance of much larger models while requiring substantially less memory and compute. The 0.85 performance on Italian SQuAD represents only a marginal decrease from the 560M-parameter large variant's 0.86, suggesting that the base model captures most of the cross-lingual capability at a fraction of the computational cost. Our evaluation demonstrates that multilingual models substantially outperform Italian-specific alternatives in absolute terms. Further fine-tuning of those multilingual models on Italian-specific datasets may capture certain language-specific patterns or cultural references that multilingual models trained predominantly on English content might miss, though our evaluation did not specifically assess this dimension. The findings from this multilingual benchmarking study directly informed the model selection decisions for the Altilia platform. Based on our evaluation, embed-multilingual-v3.0 was adopted as proprietary embedding model for cross-lingual retrieval tasks, providing consistent performance across Italian and English content. While the E5 multilingual family is adopted as open-weights multilingual alternatives for applications where privacy and costs are critical. For specialized English applications where maximum precision is required, the platform maintains the option to deploy BGE models, implementing the hybrid strategy suggested by our findings.

3.2.2 Fine-tuning Semantic Models

The multilingual benchmarking study presented in Section 3.2.1 revealed a fundamental limitation of existing embedding models: even state-of-the-art multilingual models exhibit significant performance degradation when applied to specialized

domains. The embed-multilingual-v3.0 model, for instance, experienced a 60% relative performance loss when moving from general domain content (SQuAD) to specialized technical domains (NFCorpus). This observation, consistent across multiple model families and evaluation metrics, motivated the development of a comprehensive framework for domain-adaptive fine-tuning of semantic models.

The challenge of domain adaptation in neural information retrieval is multi-faceted. Specialized domains such as Environmental, Social, and Governance (ESG) reporting, legal documentation, financial filings, and technical engineering corpora present unique characteristics that distinguish them from the general web text on which most embedding models are pre-trained. These domains exhibit specialized vocabulary and terminology that may not be well-represented in general pre-training corpora, complex document structures that require understanding of hierarchical relationships and cross-references, domain-specific semantic relationships where terms may have meanings distinct from their general usage, and multilingual content that must be processed consistently across languages. Furthermore, creating high-quality annotated datasets in these specialized domains is prohibitively expensive, as annotation requires domain expertise that is both scarce and costly. This scarcity of labeled training data represents a fundamental bottleneck for supervised domain adaptation approaches.

To address these challenges, we developed a comprehensive framework that integrates synthetic data generation with contrastive learning to enable effective domain adaptation without manual annotation. The framework, which we term Synthetic Augmentation for Guided Embeddings (SAGE), provides a scalable and reproducible methodology for adapting general-purpose embedding models to specialized retrieval contexts. In the following sections, we describe our specific implementation of the contrastive learning approach introduced in Section 3.1.2, present the synthetic data generation pipeline, detail the training and evaluation methodology, and report experimental results demonstrating the effectiveness of domain-specific fine-tuning.

Contrastive Learning Implementation

Building upon the contrastive learning foundations and bi-encoder architectures introduced in Section 3.1.2, we designed a training methodology specifically optimized for domain adaptation in low-resource settings. While the MNR loss formulation (Equation 3.3) provides the theoretical basis for our approach, effective domain adaptation requires careful consideration of training dynamics, particularly regarding batch size and learning rate scheduling. A fundamental challenge in

contrastive learning is that effectiveness scales with batch size: larger batches provide more diverse in-batch negatives, leading to better gradient estimates and improved representation quality. However, GPU memory constraints limit the batch sizes that can be processed during training, particularly for large transformer models. With standard training on consumer-grade hardware, batch sizes are typically limited to 8-32 examples, which may not provide sufficient negative diversity for learning fine-grained semantic distinctions in specialized domains.

To address this limitation, we adopt Cached Contrastive Learning [24], a technique that enables training with effectively large batch sizes while maintaining manageable memory requirements. The key insight is that gradient computation can be decomposed into two phases: a forward pass that computes embeddings, and a backward pass that computes gradients. By caching the embeddings from multiple forward passes before performing the backward pass, we can simulate the effect of a large batch without holding all intermediate activations in memory simultaneously.

The Cached-MNR procedure operates as follows. In the accumulation phase, for each sub-batch of size b, we perform a forward pass through the encoder and cache the resulting embeddings with gradient tracking disabled. This is repeated for k sub-batches, accumulating a cache of $B = k \cdot b$ embeddings. In the loss computation phase, we compute the contrastive loss using the full set of cached embeddings, treating all B embeddings as a single large batch. Finally, in the gradient computation phase, we perform the backward pass, which now benefits from the larger effective batch size.

This technique enables training with effective batch sizes of 1024 or larger on a single GPU, compared to typical limits of 8-32 for standard training. The trade-off is increased training time due to the multiple forward passes, but our experiments demonstrate that the improved representation quality from larger effective batches more than compensates for this overhead. Specifically, moving from standard MNR (batch size 10) to Cached-MNR (effective batch size 1024) yields consistent improvements of 2-3 percentage points across retrieval metrics, validating the importance of negative diversity for domain adaptation. Among the embedding models evaluated in Section 3.2.1, we selected multilingual-E5-base as the foundation for domain adaptation. This choice is motivated by several considerations that emerged from our benchmarking analysis.

First, the E5 model family employs prefix-based input formatting, using distinct prefixes (`query:` and `passage:`) to explicitly encode the asymmetric nature of the query-document relationship. This design choice reduces distributional mismatch between training and inference, as the model learns to produce

appropriate representations based on the input role. During fine-tuning and inference, all queries are prefixed with `query:` and all passages with `passage:`, ensuring consistent behavior across the training and deployment phases.

Second, multilingual-E5-base offers a favorable balance between model capacity and computational efficiency. With 278 million parameters and a maximum sequence length of 512 tokens, the model is large enough to capture complex semantic relationships while remaining tractable for fine-tuning on modest hardware. Our benchmarking results showed that simply scaling model size (e.g., moving from E5-base to E5-large) does not address domain-specific performance gaps, suggesting that targeted adaptation is more effective than increased capacity.

Third, the model's multilingual pre-training on diverse corpora covering over 100 languages makes it particularly well-suited for domains with multilingual content. ESG reporting, our primary evaluation domain, involves documents in multiple languages (67% English, 33% Italian in our corpus) that must be processed consistently. Starting from a multilingual foundation enables the fine-tuned model to maintain cross-lingual capabilities while adapting to domain-specific semantics.

Synthetic Data Generation for Training

A fundamental challenge in domain-adaptive fine-tuning is the scarcity of labeled training data. Creating high-quality query-passage pairs for contrastive training requires domain expertise to formulate realistic queries and identify relevant passages—a process that is both time-consuming and expensive. This bottleneck is particularly acute in specialized domains where annotators must possess both linguistic skills and domain knowledge, a combination that is often scarce.

To address this limitation, we developed a scalable synthetic data generation pipeline that leverages large language models (LLMs) to automatically generate high-quality training pairs from unlabeled domain corpora. The key insight is that modern LLMs, when properly prompted, can generate realistic questions that would be naturally answered by a given passage, effectively creating query-passage pairs without human intervention. This approach, which builds on the Generative Pseudo Labeling (GPL) paradigm [85], enables domain adaptation at scale with no human-in-the-loop annotation.

The synthetic data generation pipeline operates in three stages: document preprocessing and chunking, question generation using prompted LLMs, and dataset construction with quality filtering. Each stage is designed to maximize the quality and diversity of the resulting training data while maintaining computational

efficiency.

The first stage of the pipeline transforms raw documents into semantically coherent text chunks suitable for retrieval. This preprocessing step is critical because the quality of chunk boundaries directly affects both training data quality and retrieval effectiveness. Chunks that are too short may lack sufficient context for meaningful retrieval, while chunks that are too long may contain multiple distinct topics that confuse the retrieval model.

Raw documents, typically in PDF format, are first parsed to extract textual content while preserving structural information such as section boundaries, paragraph breaks, and page numbers. For documents with complex layouts, we employ Document Layout Analysis techniques to identify logical reading order and segment visual elements (tables, figures, headers) from body text. This structural awareness is particularly important for specialized domains where documents often contain heterogeneous content types.

The extracted text is then segmented into overlapping chunks using a sliding window approach. Let $D = (s_1, s_2, \ldots, s_m)$ denote a document represented as a sequence of sentences. We define a chunk c as a contiguous subsequence of sentences:

$$c_{i,w} = (s_i, s_{i+1}, \ldots, s_{i+w-1}) \tag{3.12}$$

where i is the starting position and w is the window size (number of sentences). To preserve semantic continuity across chunk boundaries, we apply overlap by setting the stride $\delta < w$, producing chunks:

$$\mathcal{C}(D) = \{c_{1,w}, c_{1+\delta,w}, c_{1+2\delta,w}, \ldots\} \tag{3.13}$$

The overlap ensures that information spanning chunk boundaries is captured in at least one chunk, reducing the risk of fragmenting relevant content. In practice, we found that a window size of approximately 3-5 sentences with 50% overlap provides a good balance between context preservation and retrieval granularity.

After segmentation, chunks are filtered based on character length to remove edge cases. Chunks shorter than 150 characters typically lack sufficient content for meaningful retrieval and may generate low-quality synthetic questions. Chunks longer than 2,048 characters may exceed the context window of the question generation model and often contain multiple distinct topics. The filtered chunks are deduplicated using SHA-256 hashing to remove exact duplicates that arise from overlapping windows or repeated content across documents. The core of the synthetic data generation pipeline is the use of large language models to generate questions for each document chunk. For each chunk c_i, we prompt an LLM to

generate a question q_i such that q_i can be answered directly and explicitly by the information contained in c_i. This framing ensures that the generated question-chunk pairs represent valid positive examples for contrastive training.

The question generation process employs few-shot in-context learning, where the LLM is provided with a small number of high-quality example question-chunk pairs from the target domain before generating questions for new chunks. This approach enables the model to learn the appropriate style, complexity, and domain-specific terminology without explicit fine-tuning. The few-shot examples are carefully curated to represent the diversity of question types and topics expected in the domain.

The generation prompt is structured as follows:

1. **Task instruction**: A clear description of the task, specifying that the model should generate a question that can be answered using only the information in the provided passage.

2. **Few-shot examples**: 3-5 high-quality question-passage pairs from the target domain, demonstrating the expected output format and quality.

3. **Target chunk**: The document chunk for which a question should be generated.

4. **Generation prompt**: A prompt requesting the model to generate a question following the pattern of the examples.

The quality of synthetic questions depends critically on prompt design. Effective prompts instruct the model to generate questions that require information contained within the chunk to answer (ensuring relevance), reflect the types of queries domain users would naturally formulate (ensuring realism), vary in complexity and specificity to provide diverse training signals (ensuring diversity), and preserve domain-specific terminology and concepts (ensuring domain alignment).

We support both commercial API-based models (such as GPT-4) and open-source alternatives (such as LLaMA 3.1 8B) for question generation. The choice depends on deployment constraints: commercial APIs offer higher quality but raise data governance concerns for sensitive domains, while open-source models can be deployed locally under full data control. Our experiments indicate that both approaches produce effective training data, with commercial models providing marginal quality improvements that may not justify the additional cost and privacy considerations for many applications.

The synthetic data generation process is distributed across multiple workers to support large-scale generation. Each worker independently processes a subset of chunks, generating questions and storing results in a distributed database. This parallelization enables processing of millions of chunks within practical time frames, making the approach scalable to enterprise-scale document collections. The raw synthetic data undergoes several filtering and organization steps to produce the final training dataset. Each generated (q_i, c_i^+) pair is treated as a potential training instance, but not all generated pairs are equally useful for training.

Quality filtering removes low-quality examples that could introduce noise into training. We apply several heuristics to identify problematic pairs: questions that are too short (fewer than 5 words) or too long (more than 50 words) are removed, as they often represent degenerate outputs. Questions that simply quote text from the chunk verbatim are filtered, as they do not represent realistic user queries. Questions containing generation artifacts (such as repeated phrases or formatting errors) are excluded.

To prevent data leakage and ensure robust evaluation, we partition the dataset such that all pairs derived from a given source document are allocated to either the training split or the development split, but never both. This document-level splitting strategy prevents the model from learning document-specific patterns (such as writing style or structural conventions) that would artificially inflate development set performance but not generalize to new documents.

The final dataset is organized into training (`synth-train`, typically 80% of documents) and development (`synth-dev`, 20% of documents) splits. The development split is used for hyperparameter tuning and checkpoint selection during training, providing an unbiased estimate of generalization performance.

Domain-Specific Model Training

The domain adaptation framework implements a modular, scalable architecture designed for both experimental flexibility and production deployment. The overall system is decomposed into three primary pipelines: (i) synthetic data generation and model training, (ii) offline evaluation on held-out benchmarks, and (iii) real-time inference for production deployment. Each pipeline is designed as an independent subsystem that can be developed, tested, and scaled independently. The training pipeline converts the synthetic dataset into a fine-tuned bi-encoder model optimized for the target domain. The pipeline implements the following workflow:

1. **Data loading**: The synthetic training pairs are loaded from the database

and organized into batches. Each batch contains query-passage pairs with minimal overlap in source documents to maximize negative diversity.

2. **Model initialization**: The bi-encoder is initialized from pre-trained weights, typically multilingual-E5-base. All model parameters are trainable, enabling full adaptation to the target domain.

3. **Training loop**: For each batch, we compute embeddings for all queries and passages, calculate the MNR loss, and update model parameters via backpropagation. When using Cached Contrastive Learning, multiple sub-batches are accumulated before the backward pass.

4. **Checkpoint selection**: Model checkpoints are saved periodically and evaluated on the development split. The checkpoint with the best performance on the target metric (typically MRR@5) is selected for deployment.

The training configuration is optimized through extensive hyperparameter search. We use the AdamW optimizer [50] with a learning rate of 1×10^{-5} and momentum parameters $\beta = (0.9, 0.999)$. The softmax temperature is set to $\tau = 0.05$, which we found to provide effective gradient signal without over-emphasizing the hardest negatives.

Learning rate scheduling employs a linear warmup followed by linear decay. The warmup phase gradually increases the learning rate from zero to the target value over a specified fraction of training steps, preventing large gradient updates early in training that could destabilize the pre-trained representations. We experimented with warmup ratios of 0.1 and 0.9, finding that larger warmup phases (0.9) yield slightly better final performance, suggesting that gradual adaptation is preferable to aggressive early updates.

For standard MNR training with batch size 10, we train for 20,000 steps, which corresponds to multiple passes through the synthetic dataset. For Cached-MNR training with effective batch size 1024, we train for only 250 steps due to the increased computational cost per step, but achieve comparable or superior performance due to the improved gradient estimates from larger batches. Following the conventions of the E5 model family, all inputs are prefixed with role-specific tokens during training. Queries are prefixed with `query:` and passages with `passage:`, ensuring that the model learns role-appropriate representations that generalize correctly at inference time. The evaluation pipeline rigorously assesses the effectiveness of the fine-tuned retriever on held-out benchmark datasets that were not used during training or checkpoint selection. This evaluation provides an

unbiased estimate of the model's generalization performance on realistic retrieval tasks. Documents in the evaluation benchmark are processed using the same preprocessing pipeline employed during training to ensure consistency in chunk extraction. This consistency is critical: differences in chunking between training and evaluation could confound the assessment of model quality with artifacts of preprocessing variation. For each test query q, the trained bi-encoder encodes q into a dense embedding. This query embedding is compared against the embeddings of all chunks in the evaluation corpus using cosine similarity. The top-k most similar chunks are retrieved as predictions, where k is a parameter chosen based on the downstream application requirements. Performance is measured using the standard IR metrics introduced in Section 3.1.5: NDCG@k, MAP@k, Recall@k, Precision@k, and MRR@k. Following the evaluation standards of MTEB and MMTEB benchmarks [53], we set $k = 5$ to focus on top-ranked results, which aligns with typical user behavior in interactive retrieval scenarios where users examine only the first few results. The combined use of these metrics provides a comprehensive view of retrieval quality: Recall captures coverage, Precision captures accuracy, MRR captures early ranking quality, and NDCG provides a unified measure that balances relevance and position. The inference pipeline enables real-time, low-latency semantic search over document collections in production settings. This component is designed to support interactive user queries in practical applications such as compliance auditing, sustainability analysis, corporate research workflows, and question answering systems. In the preprocessing phase, newly ingested documents are processed through the same ETL pipeline employed during training and evaluation. Each document is segmented into overlapping textual chunks using identical parameters (window size, overlap, length filtering) to ensure consistency with the trained model's expectations. These chunks are encoded into dense vector representations using the fine-tuned bi-encoder, producing 768-dimensional embeddings for each chunk. The resulting embeddings are stored in a dedicated vector database optimized for efficient nearest-neighbor retrieval at scale. We employ production-grade vector databases such as Milvus, Pinecone, or Weaviate that support approximate nearest neighbor search using algorithms like HNSW. These systems provide sub-millisecond query latency even over millions of vectors, satisfying the responsiveness requirements of interactive applications. At inference time, users submit free-form natural language queries, which are prefixed with `query:` and encoded into dense embeddings using the same bi-encoder. The query embedding is used to perform approximate similarity search against the precomputed chunk embeddings in the vector database. The system retrieves the top-k chunks with the highest cosine similarity to the

query vector, returning them as candidate passages for downstream processing (such as answer generation in a RAG pipeline). This architecture supports several operational requirements critical for production deployment. High throughput is achieved because query encoding and similarity search are computationally efficient operations that can handle thousands of queries per second. Continuous ingestion is supported because new documents can be added to the index without retraining, as the encoding function is fixed after fine-tuning. Horizontal scaling is enabled because the vector database can be distributed across multiple nodes for larger corpora. Finally, low latency is maintained because approximate nearest neighbor search provides sub-millisecond retrieval even at scale.

Experimental Validation

To validate the effectiveness of domain-specific fine-tuning, we conducted comprehensive experiments on an ESG (Environmental, Social, and Governance) document retrieval task. The ESG domain was selected as a representative case study because it exemplifies the challenges that motivate domain adaptation: complex heterogeneous documents with specialized terminology, multilingual content, and scarcity of labeled training data. The training corpus comprised 1,427 ESG-related documents sourced from publicly available corporate sustainability reports, regulatory filings, and industry publications. The corpus exhibits linguistic diversity, with 67% of documents written in English and 33% in Italian, reflecting the multilingual nature of ESG reporting in European markets. Document processing using the chunking pipeline described in Section 3.2.2 yielded approximately 2.9 million textual chunks. Deduplication using SHA-256 hashing reduced this to 1.5 million unique chunks, removing exact duplicates that arose from overlapping windows and repeated boilerplate content across documents. Length filtering, retaining only chunks between 150 and 2,048 characters, produced a refined corpus of approximately 410,000 chunks suitable for synthetic question generation. For controlled experimentation and computational tractability, we randomly sampled a subset of approximately 30,000 chunks from this filtered pool. Statistical analysis of the sampled subset revealed an average character length of 637.9, a mean word count of 108.5, and an average of 3.7 sentences per chunk. These statistics indicate that chunks contain sufficient context for meaningful retrieval while remaining within the model's effective context window. Synthetic question generation using LLaMA 3.1 8B produced 95,697 question-chunk pairs. Document-level splitting allocated pairs to training (80%) and development (20%) splits, ensuring that no document appears in both splits. Analysis of the synthetic dataset revealed 94,539

unique queries, indicating high lexical and semantic diversity with minimal duplication. The query-chunk alignment is nearly bijective, with each query linked to an average of 1.012 relevant chunks, reflecting a well-defined relevance structure suitable for contrastive training. Model performance is evaluated on a proprietary benchmark dataset, which we denote `esg-test`, that was manually annotated by domain experts independently of the synthetic data generation process. This separation ensures that evaluation measures genuine generalization rather than overfitting to artifacts of the synthetic data generation. The `esg-test` benchmark comprises 27 ESG-related documents from organizations not represented in the training corpus, segmented into a total of 20,662 text chunks using the same preprocessing pipeline. Relevance annotations were produced with respect to a curated set of 37 test queries designed to reflect realistic information needs in ESG analysis, covering topics such as carbon emissions, workforce diversity, governance structures, and regulatory compliance. Domain experts reviewed the corpus and identified 445 chunks as relevant to at least one test query. The dataset exhibits high diversity, with each document contributing an average of 23.4 relevant chunks, reflecting the complexity and topical breadth typical of ESG reporting. This relatively sparse relevance structure (approximately 2% of chunks are relevant to any given query) represents a challenging retrieval scenario that tests the model's ability to discriminate between relevant and irrelevant content. To contextualize the contribution of domain-specific fine-tuning, we compare against both proprietary and open-source baseline models evaluated in Section 3.2.1. These baselines represent the current state of the art in text embedding and provide a reference for assessing the value added by domain adaptation. Among proprietary baselines, we evaluate three models from OpenAI's embedding family: text-embedding-ada-002, text-embedding-3-large, and text-embedding-3-small. Among open-source baselines, we include GTE-base (305M parameters), BGE-m3 (567M parameters), and the multilingual-E5 family at three scales: E5-small (118M), E5-base (278M), and E5-large (560M). This selection enables analysis of both cross-vendor performance and scaling effects within model families. All baseline models are evaluated in a zero-shot setting without any domain-specific fine-tuning, representing the out-of-the-box performance that practitioners would observe when deploying these models on ESG retrieval tasks. This comparison directly tests our hypothesis that targeted domain adaptation can outperform both larger models and proprietary systems. We train four variants of the fine-tuned model to analyze the effects of training configuration choices:

- **MNR with warmup ratio 0.1**: Standard MNR loss with minimal learning

rate warmup (10% of training steps).

- **MNR with warmup ratio 0.9**: Standard MNR loss with extended warmup (90% of training steps).
- **Cached-MNR with warmup ratio 0.1**: Cached contrastive loss (effective batch size 1024) with minimal warmup.
- **Cached-MNR with warmup ratio 0.9**: Cached contrastive loss with extended warmup.

All fine-tuned models are initialized from multilingual-E5-base (278M parameters) and trained on the synthetic dataset described above. This controlled comparison isolates the effects of loss function and learning rate schedule while holding model architecture and training data constant. Table 3.6 presents a comprehensive comparison of fine-tuned models against baselines across multiple retrieval metrics on the `esg-test` benchmark.

Table 3.6: Performance comparison on the ESG retrieval benchmark. Fine-tuned models are initialized from multilingual-E5-base (278M parameters). WR denotes warmup ratio. Bold values indicate best performance for each metric.

Model	**Params**	**NDCG@5**	**MAP@5**	**Recall@5**	**Prec@5**	**MRR@5**
Fine-tuned models						
Cached-MNR, WR=0.9	278M	**0.498**	**0.444**	0.631	**0.142**	**0.471**
Cached-MNR, WR=0.1	278M	0.487	0.429	**0.633**	**0.142**	0.456
MNR, WR=0.9	278M	0.470	0.420	0.594	0.133	0.446
MNR, WR=0.1	278M	0.454	0.400	0.588	0.133	0.424
Proprietary baselines						
OpenAI-Ada-002	—	0.458	0.395	0.611	0.138	0.429
OpenAI-TE3-Large	—	0.446	0.384	0.598	0.135	0.414
OpenAI-TE3-Small	—	0.414	0.351	0.571	0.130	0.378
Open-source baselines						
BGE-m3	567M	0.455	0.400	0.596	0.133	0.419
GTE-base	305M	0.414	0.359	0.555	0.127	0.381
multi-E5-Large	560M	0.383	0.330	0.512	0.115	0.356
multi-E5-Base	278M	0.385	0.327	0.532	0.121	0.351
multi-E5-Small	118M	0.340	0.290	0.468	0.106	0.309

The experimental results reveal several important findings that have both theoretical and practical implications for domain-adaptive retrieval.

Synthetic supervision enables effective domain adaptation. The best fine-tuned model (Cached-MNR with WR=0.9) achieves an NDCG@5 of 0.498, representing a +4.0 percentage point absolute improvement over the strongest proprietary baseline (OpenAI-Ada-002 at 0.458). This improvement is statistically significant and practically meaningful, corresponding to substantially better ranking of relevant documents. The fine-tuned model also outperforms all open-source baselines by even larger margins, with improvements exceeding 10 percentage points over the unfine-tuned E5-base model from which it was initialized.

These results demonstrate that targeted domain adaptation through synthetic data generation can yield specialized models that generalize effectively within structured document corpora. The synthetic supervision approach successfully captures domain-specific semantics without requiring any manual annotation, validating the scalability of the methodology.

Domain adaptation outperforms model scaling. A particularly striking finding is that the fine-tuned 278M parameter model outperforms substantially larger baselines. BGE-m3 (567M parameters) achieves only 0.455 NDCG@5, while multi-E5-Large (560M parameters) achieves 0.383—both substantially below the fine-tuned model's 0.498 despite having roughly twice the parameters.

This observation suggests that performance gains in domain-specific retrieval stem primarily from alignment with domain semantics rather than raw model capacity. Simply scaling model size without domain adaptation yields diminishing returns, while targeted fine-tuning on domain-relevant data produces substantial improvements even with modest model sizes. This finding has important practical implications: organizations can achieve state-of-the-art domain-specific retrieval without the computational costs associated with deploying very large models.

Training dynamics affect performance. The comparison across fine-tuned variants reveals consistent patterns in training configuration effects. Models trained with Cached-MNR consistently outperform those using standard MNR, with improvements of 2-3 percentage points across metrics. This confirms that the larger effective batch sizes enabled by gradient caching produce better gradient estimates and more robust representations.

The warmup ratio also influences final performance, with extended warmup (WR=0.9) yielding slightly better results than minimal warmup (WR=0.1). This suggests that gradual adaptation from pre-trained weights is preferable to aggressive early updates, allowing the model to preserve useful general representations while adapting to domain-specific patterns.

Environmental efficiency. Beyond performance metrics, the fine-tuning approach demonstrates favorable computational and environmental characteristics. Training the best model (Cached-MNR, WR=0.9) required approximately 0.256 kg CO_2-equivalent emissions, estimated from energy consumption using standard conversion factors. This is orders of magnitude lower than pre-training a model from scratch, demonstrating that effective domain adaptation can be achieved through efficient fine-tuning of existing foundational models.

Model Lifecycle Management

Deploying domain-adapted embedding models in production environments requires robust model lifecycle management practices that ensure reliability, reproducibility, and continuous improvement. The transition from experimental fine-tuning to production deployment introduces operational challenges that must be addressed through systematic engineering practices. All model artifacts are versioned and tracked using MLflow, an open-source platform for managing the machine learning lifecycle. Each training run generates a unique experiment identifier that is linked to the complete specification of the run: dataset version (identified by content hash), hyperparameter configuration, random seed, hardware environment, and software dependencies. This comprehensive tracking ensures that any experimental result can be reproduced exactly, which is essential for debugging, auditing, and regulatory compliance. Model registries maintain the lineage of deployed models, recording the relationship between production models and their training artifacts. When a new model version is deployed, the registry records which training run produced it, what evaluation metrics it achieved, and who approved the deployment. This audit trail enables rollback to previous versions if performance regressions are detected in production, and supports compliance with governance requirements in regulated industries. The deployment pipeline implements continuous integration practices adapted for machine learning systems. Before any model can be considered for deployment, it must pass a suite of automated checks. Unit tests verify that the model produces embeddings of the expected dimension and that similarity scores fall within valid ranges. Integration tests confirm that the model integrates correctly with the inference pipeline, including preprocessing, encoding, and vector database operations. Performance tests evaluate the model on held-out benchmarks and compare against the currently deployed model and historical baselines. Deployment gates enforce minimum performance thresholds: a new model must exceed the current production model on primary metrics (NDCG@5, MRR@5) by a statistically significant margin before

approval. These gates prevent regression by ensuring that each deployed model represents a genuine improvement over its predecessor. New models are deployed using gradual rollout strategies that limit exposure while gathering production metrics. Initially, a small percentage of traffic (typically 5-10%) is routed to the new model while the remainder continues to use the incumbent. Key performance indicators are monitored, including retrieval latency (to detect computational regressions), user engagement metrics (such as click-through rates on retrieved documents), and downstream task performance (such as answer quality in RAG applications). Statistical hypothesis testing determines when a new model variant can safely replace the incumbent. We use sequential analysis methods that enable early stopping when the evidence for improvement (or regression) becomes conclusive, minimizing the duration of suboptimal user experience while maintaining statistical rigor.

Continuous Learning from User Feedback

Production deployment generates valuable feedback signals that can be leveraged to improve model performance over time. We implement a continuous learning framework that collects, curates, and incorporates user feedback into the model training pipeline, creating a virtuous cycle of improvement. User interactions with the retrieval system provide implicit signals about result quality without requiring explicit annotation effort. Several interaction patterns are informative:

Click-through behavior indicates which retrieved documents users find relevant. Documents that receive clicks from multiple users for similar queries are likely relevant, while documents that are consistently skipped may be false positives. However, click data must be interpreted carefully due to position bias: users are more likely to click on higher-ranked results regardless of relevance.

Dwell time on retrieved documents provides a signal of document utility. Documents where users spend substantial time are likely providing value, while documents that are quickly abandoned may be irrelevant or low-quality. Long dwell times followed by successful task completion (such as submitting an answer based on the document) are particularly strong positive signals. Query reformulation patterns indicate retrieval failures. When users submit a query, receive results, and then submit a modified query on the same topic, this suggests that the initial results did not satisfy their information need. The relationship between original and reformulated queries can reveal semantic gaps in the retrieval model's understanding. When available, explicit relevance judgments from domain experts provide high-quality training signals that complement implicit feedback. The

framework supports annotation interfaces that enable experts to label retrieved chunks as relevant, partially relevant, or irrelevant with respect to specific queries. These annotations are integrated into the training pipeline using a hybrid approach that combines expert labels with synthetic data. The loss function is modified to weight expert-annotated pairs more heavily than synthetic pairs, reflecting their higher reliability:

$$\mathcal{L}_{\text{hybrid}} = \lambda \mathcal{L}_{\text{expert}} + (1 - \lambda)\mathcal{L}_{\text{synthetic}} \tag{3.14}$$

where λ controls the relative weight of expert annotations. In practice, we find that even small amounts of expert annotation ($\lambda = 0.1$ to 0.2) can meaningfully improve performance, particularly for challenging queries where synthetic supervision may be noisy. To maximize the value of limited expert annotation budget, the framework implements active learning strategies that select the most informative examples for human review. Rather than annotating random samples, active learning prioritizes examples where annotation would most improve model performance. Uncertainty sampling identifies query-chunk pairs where the model exhibits high prediction uncertainty—cases where the model assigns similar scores to the top candidates rather than confidently ranking one above the others. These uncertain cases often represent the boundary between relevant and irrelevant content, where human judgment can provide the most informative signal. Diversity sampling ensures that selected examples cover the breadth of the domain rather than concentrating on a narrow subset of topics or document types. By maintaining diversity in the annotation queue, we ensure that model improvements generalize across the full range of retrieval scenarios. The combination of uncertainty and diversity sampling achieves greater model improvement per annotation than random sampling, enabling effective adaptation even with limited expert availability. The continuous learning pipeline supports periodic model retraining that incorporates newly collected feedback. Retraining can be triggered by several conditions: scheduled intervals (such as weekly or monthly), accumulated feedback volume exceeding a threshold, or detected performance degradation in production metrics. Each retraining cycle produces a new model candidate that undergoes the standard evaluation and deployment workflow described in Section 3.2.2. This ensures that feedback-driven updates maintain or improve retrieval quality, preventing degradation from noisy feedback signals. The integration of domain-adaptive fine-tuning with continuous learning creates a self-improving system where production deployments generate training signals that enable progressive model improvement. This framework has proven essential for maintaining retrieval quality as document collections evolve and user information needs shift.

Chapter 4

Question Answering and Large Language Models

4.1 State of the Art

Question Answering (QA) represents one of the core capabilities of modern Natural Language Processing systems, enabling users to extract precise information from large collections of documents using natural language queries. This functionality has undergone a fundamental transformation over the past decade, evolving from extractive approaches that identify answer spans within source documents to generative systems capable of synthesizing comprehensive responses from multiple sources. This section reviews the evolution of QA systems, examines the emergence of prompting techniques and in-context learning as alternatives to traditional fine-tuning, and discusses the trade-offs between these paradigms in practical applications.

4.1.1 Evolution of QA: From Extractive to Generative

Traditional QA systems followed an extractive paradigm, where the task was formulated as identifying specific spans of text within a given passage that directly answer a query. This approach was popularized by benchmark datasets such as SQuAD (Stanford Question Answering Dataset) [71], which comprises over 100,000 question-answer pairs derived from Wikipedia passages. A subsequent

release, SQuAD 2.0 [70], introduced an additional 50,000 unanswerable questions designed to evaluate systems' ability to identify when no answer exists in the given passage. The introduction of BERT (Bidirectional Encoder Representations from Transformers) [16] revolutionized extractive QA by enabling bidirectional pre-training through masked language modeling (MLM), allowing models to simultaneously consider context from both directions and produce richer contextual representations. The extractive paradigm proved particularly effective for reading comprehension tasks where answers are explicitly stated within the source text. BERT and its variants including RoBERTa [49], which optimized the pre-training procedure, and ALBERT [39], which introduced parameter-sharing techniques for efficiency achieved remarkable performance on extractive benchmarks. These models were typically fine-tuned on specific QA datasets by adding a classification layer to predict the logits corresponding to the start and end positions of the answer span within the input passage. However, extractive QA presents inherent limitations. First, it relies on the strict assumption that the answer exists as a contiguous span within the provided context a condition often violated by complex questions requiring synthesis or multi-hop inference. Second, when dealing with multiple documents or long passages, extractive systems struggle to aggregate information from disparate sources. Third, the output is constrained to exact textual reproduction, limiting the system's ability to paraphrase, summarize, or adapt the response tone to the user's needs. The emergence of large-scale generative language models fundamentally shifted the QA paradigm. The GPT series, beginning with GPT [67] and evolving through GPT-2 [68], GPT-3 [7], and GPT-4 [1], demonstrated that decoder-only architectures trained autoregressively on massive corpora could perform QA without explicit fine-tuning. GPT-3's 175 billion parameters showcased remarkable few-shot learning capabilities, enabling the model to perform tasks by conditioning on just a few examples provided in the prompt. Generative QA systems offer several advantages over their extractive counterparts. They can synthesize information from multiple sources, produce natural-sounding responses that may not appear verbatim in the source text, and handle questions that require reasoning, inference, or world knowledge beyond the provided context. More recent generative models like GPT-3 and GPT-4 and also open weights models like Llama [83] have further advanced generative QA with instruction tuning that increase the ability of those model to follow user instructions. Nevertheless, generative models introduce new challenges, most notably the phenomenon of hallucination generating plausible but factually incorrect responses. This limitation has motivated the development of Retrieval-Augmented Generation (RAG) systems [42], which combine retrieval mechanisms with generative models to

ground responses in retrieved evidence and to allow the generative models the access to updated and domain specific knowledge.

4.1.2 Prompting Techniques and In-Context Learning

The advent of large language models has given rise to a new paradigm for adapting pre-trained models to specific tasks: prompting. Rather than updating model weights through gradient descent on task-specific datasets, prompting leverages the model's pre-existing capabilities by conditioning generation on carefully crafted input text. This approach, particularly when combined with in-context learning, has proven remarkably effective for a wide range of NLP tasks, including question answering.

Zero-shot prompting represents the simplest form of this paradigm, where the model is provided with a task description and expected to perform without any examples. For instance, a QA prompt might instruct: "Answer the following question based on the provided context." While zero-shot performance varies significantly across tasks, larger models generally exhibit stronger zero-shot capabilities due to the breadth of knowledge encoded during pre-training [90].

Few-shot prompting extends this approach by including a small number of demonstration examples in the prompt. Brown et al. [7] showed that GPT-3's performance on various benchmarks improved substantially when provided with just a few examples, a phenomenon termed in-context learning. The model appears to infer the task structure and output format from these demonstrations without any parameter updates. This capability has profound implications for practical applications, as it enables rapid task adaptation without the computational overhead of fine-tuning.

Chain-of-Thought (CoT) prompting [91] represents a significant advance for tasks requiring reasoning. Rather than directly outputting an answer, CoT prompting encourages the model to generate intermediate reasoning steps before arriving at the final response. For example, in a multi-step arithmetic or logical reasoning task, the model is prompted to "think step by step," decomposing the problem into manageable sub-problems. This approach has demonstrated substantial improvements on complex QA tasks, particularly those involving numerical reasoning or multi-hop inference. Notably, CoT reasoning has been observed as an emergent capability that scales with model size, typically becoming effective in models exceeding 10 billion parameters [90]. Subsequent work has introduced variations such as zero-shot CoT [37] and self-consistency [89]. The effectiveness of prompting depends critically on prompt design often referred

to as prompt engineering. Key considerations include the clarity and specificity of instructions, the selection and ordering of few-shot examples, the format of expected outputs, and the inclusion of constraints or guardrails. Research has shown that seemingly minor variations in prompt wording can lead to significant performance differences [103], highlighting the importance of systematic prompt optimization. For QA applications, effective prompts typically include: (1) a clear role definition for the model (e.g., "You are a Question Answering system"); (2) explicit instructions about the task and expected output format; (3) the context or retrieved passages from which the answer should be derived; (4) the user's question; and (5) optional constraints such as response length or behavior when the answer cannot be determined from the context. Template engineering the systematic design and optimization of prompt templates has emerged as a crucial skill for deploying LLM-based QA systems in production. Instruction tuning has further enhanced the prompting capabilities of LLMs. Models such as InstructGPT [61], Flan-T5 [13], and the instruction-tuned variants of Llama are fine-tuned on diverse instruction-following datasets, making them more responsive to natural language instructions. This training paradigm bridges the gap between prompting and fine-tuning, producing models that are simultaneously general-purpose and instruction-following.

4.1.3 Fine-tuning vs Prompting

The choice between fine-tuning and prompting represents a fundamental decision in deploying LLMs for QA applications. Each approach offers distinct advantages and trade-offs that must be considered in the context of specific use cases, available resources, and performance requirements.

Fine-tuning involves updating some or all of a pre-trained model's parameters on a task-specific dataset. This approach typically yields the highest task performance, as the model's representations are explicitly optimized for the target domain and task. Fine-tuning is particularly advantageous when: (1) substantial labeled training data is available; (2) the target domain differs significantly from the pre-training corpus; (3) consistent, highly optimized performance is required; and (4) computational resources for training are available. However, full fine-tuning of large models presents significant challenges. Training models with billions of parameters requires substantial GPU memory and compute time. Additionally, fine-tuned models may exhibit catastrophic forgetting [36], losing general capabilities as they specialize for the target task. Storage and deployment costs also increase when maintaining multiple fine-tuned variants for different

tasks or domains.

Parameter-Efficient Fine-Tuning (PEFT) methods address these challenges by updating only a small subset of model parameters while keeping the majority frozen. Parameter-Efficient Fine-Tuning (PEFT) methods enable adaptation of large language models with minimal computational overhead.

Low-Rank Adaptation (LoRA) approximates weight updates using low-rank matrix decomposition. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA parameterizes the update as:

$$W = W_0 + \Delta W = W_0 + BA \tag{4.1}$$

where $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$, with rank $r \ll \min(d, k)$. During inference, the forward pass becomes:

$$h = W_0 x + \frac{\alpha}{r} \cdot BAx \tag{4.2}$$

where α is a scaling factor that controls the magnitude of the low-rank update.

Quantized LoRA (QLoRA) combines 4-bit quantization with LoRA for memory-efficient fine-tuning. The quantized weight W^{NF4} uses NormalFloat4 quantization with double quantization:

$$W^{\text{NF4}} = \text{doubleDequant}(c_1^{\text{FP32}}, c_2^{k\text{-bit}}, W^{\text{NF4}}) + BA \tag{4.3}$$

.

Other PEFT approaches include adapter layers [29] and BitFit [98].

Prompting, by contrast, offers immediate deployment without training, making it ideal for rapid prototyping and scenarios where labeled data is scarce. The same base model can be applied to multiple tasks simply by changing the prompt, reducing infrastructure complexity. Prompting also preserves the model's general capabilities, as no parameters are modified. The limitations of prompting include sensitivity to prompt design, the constraint of context window size, and potentially lower peak performance compared to fine-tuned models on specialized tasks. Crucially, regarding inference costs, prompting often necessitates larger context windows (to include instructions and few-shot examples), which increases the per-query cost compared to fine-tuned models that can often perform the same task with minimal context instructions. Furthermore, prompting generally requires larger base models (e.g., Llama-70B) to achieve the same reliability that a smaller fine-tuned model (e.g., Llama-8B) might deliver. In practice, a hybrid approach

often proves most effective. Instruction-tuned base models provide strong general capabilities enhanced through prompting, while PEFT techniques enable targeted adaptation for specific domains or tasks. The optimal strategy depends on the specific requirements of the application, the availability of training data, and the computational resources at hand. Table 4.1 summarizes the key trade-offs between fine-tuning and prompting approaches for QA applications.

Table 4.1: Comparison of fine-tuning and prompting approaches for QA applications.

Aspect	**Fine-tuning**	**Prompting**
Training data required	High	None/Low
Computational cost (training)	High	None
Computational cost (inference)	Low–Medium*	Medium–High*
Task-specific performance	Highest	Good
Deployment flexibility	Low	High
Domain adaptation	Excellent	Limited
General capabilities	May degrade	Preserved
Time to deployment	Days–Weeks	Minutes–Hours

* Inference cost varies by model size and context length usage.

4.1.4 Evaluation Metrics for Question Answering

Evaluating the quality of question answering systems presents unique challenges that distinguish it from other NLP tasks. Unlike classification problems with discrete labels, QA systems produce free-form textual responses where multiple valid answers may exist for a single question. A correct answer can be expressed in numerous ways—varying in length, specificity, and phrasing—making evaluation inherently more complex than simple string matching. This section presents a comprehensive taxonomy of evaluation metrics for QA systems, organized by their underlying methodology and the aspects of answer quality they capture.

Syntactic Metrics

Syntactic metrics evaluate formal response aspects through lexical overlap between generated and reference answers. While limited in their ability to capture semantic

equivalence, these metrics remain widely used due to their simplicity, reproducibility, and computational efficiency.

ROUGE. The Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [47] family of metrics was originally developed for summarization evaluation but has been widely adopted for QA assessment. ROUGE measures n-gram overlap between generated and reference texts, with several variants capturing different aspects of textual similarity.

ROUGE-N computes n-gram recall between a candidate text and a reference:

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{References}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{References}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \tag{4.4}$$

where $\text{Count}_{\text{match}}(\text{gram}_n)$ is the maximum number of n-grams co-occurring in both candidate and reference texts.

ROUGE-L employs the Longest Common Subsequence (LCS) to capture sentence-level structure similarity:

$$\text{ROUGE-L} = \frac{(1+\beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{4.5}$$

where $R_{lcs} = \frac{LCS(X,Y)}{m}$ represents recall and $P_{lcs} = \frac{LCS(X,Y)}{n}$ represents precision, with m and n being the lengths of the reference and candidate sequences respectively. The underlying intuition is that longer shared subsequences indicate greater similarity between texts.

The primary advantage of ROUGE metrics is their language independence and ease of computation. However, they suffer from significant limitations: they do not consider word semantics, treating synonyms as entirely different tokens, and are sensitive to surface-level variations in word choice and sentence structure. A semantically correct answer phrased differently from the reference may receive a low ROUGE score.

F1 Score. The token-level F1 score measures the harmonic mean of precision and recall based on word overlap between generated and reference answers:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.6}$$

where Precision is the fraction of tokens in the generated answer that appear in the reference, and Recall is the fraction of reference tokens that appear in

the generated answer. This metric is employed in prominent QA benchmarks including SQuAD [71] and TriviaQA [33].

The F1 score effectively summarizes both precision and recall in a single metric and handles class imbalance well. However, like ROUGE, it operates purely at the lexical level and cannot capture semantic equivalence between differently-phrased answers.

Semantic Metrics

Semantic metrics address the fundamental limitation of syntactic approaches by evaluating meaning equivalence rather than surface-form similarity. These metrics leverage neural language models to capture semantic relationships between generated and reference answers.

BERTScore. BERTScore [101] computes semantic similarity using contextual embeddings from pre-trained transformer models. Given a candidate sequence $\hat{x} = \langle \hat{x}_1, \ldots, \hat{x}_k \rangle$ and a reference sequence $x = \langle x_1, \ldots, x_l \rangle$, BERTScore first obtains contextual embeddings for each token using BERT or similar models. The recall component is computed as:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \tag{4.7}$$

where $\mathbf{x}_i$ and $\hat{\mathbf{x}}_j$ are the contextual embeddings of tokens x_i and $\hat{x}_j$ respectively. Precision and F1 variants are computed analogously.

BERTScore's key advantage is its ability to recognize semantic equivalence between lexically different expressions. For instance, it can identify that "the capital of France" and "Paris" convey related meanings, unlike purely lexical metrics. However, BERTScore is computationally more expensive than syntactic metrics and its performance depends on the quality and domain coverage of the underlying language model.

BEM Score. The BERT-based Evaluation Metric (BEM) [9] employs a BERT model fine-tuned specifically for answer equivalence assessment. Unlike BERTScore, which computes general semantic similarity, BEM is trained on the specific task of determining whether two answers to a question are equivalent. The model receives a question q, a candidate answer a_c, and a reference answer a_r as input,

and returns a score quantifying the probability that the candidate and reference answers are equivalent:

$$\text{BEM}(q, a_c, a_r) = P(\text{equivalent}|q, a_c, a_r) \tag{4.8}$$

Reference-Free Metrics

A fundamental challenge in evaluating QA systems is the scarcity of gold-standard reference answers, particularly in domain-specific applications where expert annotation is expensive and time-consuming. Reference-free metrics address this challenge by leveraging large language models to assess answer quality without requiring ground-truth responses. These metrics have gained prominence with the development of frameworks such as RAGAS [19] and TruLens.

Answer Relevance. Answer relevance measures whether the generated response appropriately addresses the query. This metric assesses the semantic alignment between the question asked and the answer provided, regardless of factual correctness. A common implementation generates hypothetical questions from the answer and measures their similarity to the original query:

$$\text{Answer Relevance} = \frac{1}{k}\sum_{i=1}^{k} \text{sim}(q, q_i) \tag{4.9}$$

where q is the original question, q_i are k questions generated from the answer, and $\text{sim}(\cdot, \cdot)$ is a similarity function (typically cosine similarity of embeddings). High answer relevance indicates that the response addresses what was asked, though it does not guarantee factual accuracy.

Groundedness. Groundedness (also termed faithfulness) assesses the degree to which the generated answer is supported by the provided context or source documents. This metric is particularly critical for knowledge-intensive QA where answers should be derived from retrieved evidence rather than the model's parametric knowledge. Groundedness is typically computed by identifying claims in the generated response and verifying whether each claim can be inferred from the source context:

$$\text{Groundedness} = \frac{|C_{\text{supported}}|}{|C_{\text{total}}|} \tag{4.10}$$

where $C_{\text{supported}}$ is the set of claims in the response that are supported by the context, and C_{total} is the total number of claims.

Low groundedness scores indicate potential hallucination—the generation of plausible but unsupported or fabricated information. This metric is essential for applications requiring verifiable, trustworthy responses, such as legal, medical, or financial QA systems.

Context Relevance. Context relevance evaluates whether retrieved passages (in retrieval-augmented systems) are relevant for answering the given query. While primarily an information retrieval metric, it provides important diagnostic information for QA systems that rely on retrieved context:

$$\text{Context Relevance} = \frac{|S_{\text{relevant}}|}{|S_{\text{total}}|} \tag{4.11}$$

where S_{relevant} is the set of sentences in the retrieved context deemed relevant to the query, and S_{total} is the total number of sentences. This metric helps identify whether QA failures stem from poor retrieval or poor generation.

4.2 Contributions

4.2.1 Benchmarking Chat Models for Question Answering

The proliferation of large language models has created a pressing need for systematic evaluation frameworks that can assess QA capabilities across different dimensions. While the previous sections examined the theoretical foundations of generative QA and the trade-offs between fine-tuning and prompting approaches, practical deployment decisions require empirical evidence of how different models perform on realistic QA tasks. To address this need, we conducted a comprehensive benchmarking study [60] that evaluates state-of-the-art LLMs on question answering tasks, providing actionable insights for model selection in multilingual applications.

Evaluation Methodology

Our evaluation framework was designed to assess LLM performance across multiple complementary dimensions, employing the metrics described in Section 4.1.4.

The methodology spans syntactic, semantic, and reference-free evaluation approaches, enabling a comprehensive understanding of model behavior across different quality dimensions. Models were evaluated in a context-augmented configuration where relevant passages were provided alongside each question. This setup reflects realistic deployment scenarios where QA systems have access to source documents from which answers should be derived. For each query, the top 10 most relevant passages were retrieved using Cohere's embed-multilingual-v3.0, selected based on the embedding evaluation results presented in Section 3.2.1. Documents were processed using 512-token chunks, providing sufficient context while remaining within the effective attention span of all evaluated models. To assess generalization capabilities across domains and languages, we employed a diverse set of evaluation datasets spanning general knowledge, specialized domains, and cross-lingual configurations. Table 4.2 summarizes the datasets.

Table 4.2: Overview of datasets used for QA evaluation. The selection spans general and specialized domains across English and Italian, including cross-lingual configurations.

Dataset	**Domain**	**Language**	**Task**	**Samples**
SQuAD-en	Open/General	English	QA	150
SQuAD-it	Open/General	Italian	QA	150
CovidQA	Bio-Medical	English	QA	124
NarrativeQA-Books	Narrative	English	RC	50
NarrativeQA-Movies	Narrative	English	RC	50
NarrativeQA-Books-tran	Narrative	Cross-lingual	RC	50
NarrativeQA-Movies-tran	Narrative	Cross-lingual	RC	50

SQuAD (Stanford Question Answering Dataset) serves as our primary general-domain benchmark. We used 150 parallel question-answer pairs from both the English original and the Italian translation (SQuAD-it), enabling direct cross-lingual comparison on equivalent content. The parallel nature of these datasets isolates the effect of language on QA performance. **CovidQA** provides a specialized medical domain evaluation, containing 124 question-answer pairs derived from 85 unique COVID-19 research articles. This dataset tests models' ability to handle technical vocabulary and domain-specific reasoning in a knowledge-intensive context where precision is critical. **NarrativeQA** presents a challenging reading comprehension task spanning books and movie scripts, requiring under-

standing of extended narratives and complex queries. We evaluated on a balanced subsample of 100 queries (50 from books, 50 from movie scripts). **NarrativeQA-Cross-Lingual** was created by maintaining the original English documents while translating the question-answer pairs into Italian. This configuration tests how well models can bridge the language gap between source documents and queries—a capability essential for multilingual information systems. Following the metric taxonomy presented in Section 4.1.4, we employed three complementary evaluation approaches:

- **Syntactic metrics**: ROUGE-L and F1 score to measure lexical overlap with reference answers.
- **Semantic metrics**: BERTScore and BEM to capture meaning equivalence beyond surface-form similarity.
- **Reference-free metrics**: Answer relevance, context relevance, and groundedness (implemented via TruLens with GPT-3.5-turbo) to assess quality dimensions without gold-standard answers.

Human evaluation on a subset of responses validated the reliability of automated metrics, using a 5-point Likert scale assessed by three independent annotators with standardized training. Our evaluation encompasses four large language models representing different points in the performance-efficiency spectrum. This selection enables analysis of trade-offs between model capability, computational requirements, and deployment constraints. Table 4.3 summarizes the evaluated models.

Table 4.3: Large language models evaluated for QA tasks. The selection spans from efficient open-source models to state-of-the-art proprietary systems.

Model	Provider	Type	Parameters	Context
GPT-4o	OpenAI	Proprietary	>175B	128K
Llama 3.1 8B	Meta	Open-source	8B	8K
Mistral-Nemo	MistralAI	Open-source	12B	128K
Gemma 2B	Google	Open-source	2.5B	128K

GPT-4o serves as the high-performance benchmark, representing the current state of the art in proprietary LLMs. Its inclusion establishes an upper bound for

expected performance while providing a reference point for evaluating open-source alternatives.

Llama 3.1 8B balances strong performance with open-weight flexibility, enabling fine-tuning for specific applications. This model represents an attractive option for deployments requiring both capability and customizability.

Mistral-Nemo was selected for its instruction-following reliability and reasonable computational demands. At 12 billion parameters, it occupies the middle ground between efficiency and capability.

Gemma 2B represents the ultra-efficient end of the spectrum, enabling deployment in resource-constrained environments. Its inclusion allows assessment of performance trade-offs when moving to significantly smaller models.

All models were evaluated in a zero-shot prompting configuration without task-specific fine-tuning, reflecting practical scenarios where models must generalize to new domains without additional training. The rationale behind this specific model selection is to comprehensively cover the spectrum of deployment constraints typical of industrial Intelligent Document Processing applications. GPT-4o is included to establish a theoretical upper bound for state-of-the-art proprietary performance. Llama 3.1 8B and Mistral-Nemo were selected as they represent the current ”sweet spot” for on-premise deployments, offering a critical balance between high reasoning capabilities and data privacy requirements. Finally, Gemma 2B is included to test the lower bounds of hardware requirements, simulating extreme resource-constrained environments (such as edge deployments) where inference efficiency is the primary bottleneck.

Results and Analysis

Table 4.4 presents the syntactic metric results across all models and datasets.

Table 4.4: Syntactic evaluation results (ROUGE-L / F1) for QA tasks across different domains and languages.

Model	SQuAD-en	SQuAD-it	CovidQA	NarrativeQA
GPT-4o	0.26 / 0.25	0.21 / 0.18	0.21 / 0.13	0.15 / 0.12
Llama 3.1 8B	0.72 / 0.69	0.57 / 0.54	0.22 / 0.15	0.13 / 0.11
Mistral-Nemo	0.43 / 0.41	0.27 / 0.25	0.27 / 0.17	0.27 / 0.23
Gemma 2B	0.40 / 0.39	—	0.24 / 0.16	0.16 / 0.12

The syntactic evaluation reveals notable performance variations across models.

Llama 3.1 8B demonstrates superior lexical alignment on general QA tasks, achieving ROUGE-L scores of 0.72 on SQuAD-en and 0.57 on SQuAD-it. This strong syntactic performance suggests that Llama generates responses closely mirroring reference answer phrasing. Interestingly, GPT-4o shows relatively low syntactic scores (0.26 on SQuAD-en) despite being the most capable model, indicating that it generates more elaborate or differently-phrased responses that diverge lexically from references while potentially remaining semantically correct.

A consistent degradation pattern emerges across all models when moving from general to specialized domains. CovidQA scores are substantially lower than SQuAD scores, with ROUGE-L values ranging from 0.21 to 0.27. This reflects the challenge of generating responses matching the technical terminology expected in medical contexts.

Cross-lingual performance shows larger gaps for syntactic metrics: Llama 3.1 8B drops from 0.72 (SQuAD-en) to 0.57 (SQuAD-it), while Mistral-Nemo decreases from 0.43 to 0.27. These gaps highlight the sensitivity of syntactic metrics to language-specific surface patterns. Table 4.5 presents the semantic metric results. Semantic metrics consistently show higher scores than syntactic

Table 4.5: Semantic evaluation results (BERTScore / BEM) for QA tasks.

Model	**SQuAD-en**	**SQuAD-it**	**CovidQA**	**NarrativeQA**
GPT-4o	0.85 / 0.93	0.81 / 0.92	0.85 / 0.61	0.85 / 0.48
Llama 3.1 8B	0.92 / 0.90	0.90 / 0.79	0.85 / 0.61	0.85 / 0.46
Mistral-Nemo	0.88 / 0.94	0.83 / 0.82	0.86 / 0.62	0.88 / 0.56
Gemma 2B	0.88 / 0.77	—	0.85 / 0.43	0.86 / 0.35

measures, with BERTScore ranging from 0.81 to 0.92 across tasks. This pattern confirms that models generate semantically appropriate answers even when they deviate lexically from references—validating the importance of semantic evaluation as discussed in Section 4.1.4. A notable finding is the divergence between BERTScore and BEM across domains. While BERTScore remains stable (0.85–0.88) even on specialized tasks, BEM shows pronounced domain effects, dropping from 0.77–0.94 on general tasks to 0.43–0.62 on specialized domains. This suggests that BEM's task-specific training makes it more sensitive to domain shift, potentially providing better discrimination between answer quality levels. GPT-4o achieves the highest BEM score (0.93) on SQuAD-en despite its low syntactic scores, confirming that its responses capture semantic content effectively

while using different surface forms. Mistral-Nemo achieves comparable BEM performance (0.94) and maintains the strongest scores on specialized tasks (0.62 on CovidQA, 0.56 on NarrativeQA), suggesting robust cross-domain generalization. Cross-lingual performance is more stable for semantic metrics than syntactic ones. BERTScore shows relatively small gaps between English (0.85–0.92) and Italian (0.81–0.90), indicating that semantic understanding transfers well across languages even when surface patterns differ substantially. Table 4.6 presents the LLM-based reference-free metrics.

Table 4.6: Reference-free evaluation results (Answer Relevance / Context Relevance / Groundedness) for QA tasks.

Model	**SQuAD-en**	**SQuAD-it**	**CovidQA**
GPT-4o	1.00 / 0.90 / 0.79	0.99 / 0.80 / 0.81	0.89 / 0.82 / 0.61
Llama 3.1 8B	1.00 / 0.89 / 0.67	0.99 / 0.80 / 0.71	0.86 / 0.82 / 0.62
Mistral-Nemo	1.00 / 0.89 / 0.78	0.98 / 0.81 / 0.78	0.91 / 0.82 / 0.64
Gemma 2B	0.98 / 0.90 / 0.67	—	0.77 / 0.82 / 0.51

The reference-free evaluation reveals a critical pattern: a substantial and consistent gap between answer relevance and groundedness scores across all models. Answer relevance remains high (0.98–1.0 for SQuAD-en, 0.86–0.91 for CovidQA), indicating that responses appropriately address the queries. However, groundedness scores are notably lower (0.67–0.79 for SQuAD-en, 0.51–0.64 for CovidQA), revealing that models sometimes generate plausible but unfaithful answers not fully supported by the provided context.

This answer relevance–groundedness gap represents a fundamental challenge in LLM-based QA. Models appear to prioritize generating fluent, relevant-seeming responses over strictly adhering to provided context, potentially incorporating parametric knowledge or fabricating details. The gap is remarkably consistent across models, sizes, and languages, suggesting a systematic limitation of current approaches rather than a model-specific issue.

Mistral-Nemo demonstrates the best balance, achieving 0.78 groundedness on SQuAD-en while maintaining perfect answer relevance (1.0). GPT-4o shows similar groundedness (0.79) with slightly higher answer relevance on specialized tasks. Groundedness degrades substantially on CovidQA (0.51–0.64), indicating that specialized domains pose greater challenges for maintaining factual fidelity. The cross-lingual evaluation using NarrativeQA with Italian queries over English

documents tests language transfer capabilities in a challenging configuration. Table 4.7 presents these results.

Table 4.7: Cross-lingual QA results on NarrativeQA with English documents and Italian queries.

Model	BERTScore	BEM	AR / CR / G
GPT-4o	0.83	0.46	0.95 / 0.52 / 0.38
Llama 3.1 8B	0.82	0.44	0.92 / 0.53 / 0.37
Mistral-Nemo	0.83	0.22	0.97 / 0.52 / 0.39

Cross-lingual scenarios present substantially greater challenges. BEM scores drop significantly (0.22–0.46 compared to 0.46–0.94 in monolingual settings), indicating that cross-lingual generation often produces semantically divergent responses. Groundedness is particularly affected (0.37–0.39), suggesting models struggle to remain faithful to source content when bridging language boundaries.

GPT-4o demonstrates superior cross-lingual capabilities, maintaining the highest BEM score (0.46) in the translated configuration. This advantage likely stems from its extensive multilingual pre-training and larger parameter count. To validate automated metrics, we computed Spearman correlations with human judgments on NarrativeQA subsamples (Table 4.8). BEM demonstrates strong

Table 4.8: Spearman correlations between automated metrics and human judgments.

Metric	Books	Movies
BEM	0.735	0.704
Answer Relevance	0.436	0.565

correlation with human judgment ($\rho = 0.735$ for books, $\rho = 0.704$ for movies), substantially outperforming reference-free answer relevance ($\rho = 0.436$ and $\rho = 0.565$). This validates the discussion in Section 4.1.4: while reference-free metrics offer practical advantages for scalable evaluation, reference-based semantic metrics remain more reliable proxies for human assessment. The evaluation results enable evidence-based recommendations tailored to various deployment scenarios. For applications demanding the highest accuracy across diverse domains, GPT-4o emerges as the optimal choice despite lower syntactic scores, as it achieves

the highest semantic quality and superior cross-lingual capabilities, albeit with higher costs and API dependencies. When balancing quality with the need for open-source deployment, Mistral-Nemo offers a compelling alternative. Its consistent cross-domain performance and strong groundedness make it particularly suitable for applications where both accuracy and transparency are paramount. In scenarios prioritizing high throughput and response format consistency, Llama 3.1 8B provides strong general-domain performance at a lower computational cost. Finally, for severely resource-constrained environments, Gemma 2B enables deployment despite notable trade-offs, such as performance drops on specialized tasks and a lack of Italian language support. Our benchmarking study reveals several fundamental patterns with significant implications for QA system development. A primary concern is the persistent ”groundedness gap” exhibited by all models, where answer relevance consistently outpaces actual groundedness. This phenomenon spans different architectures and sizes, suggesting that current training approaches inadvertently prioritize fluency over strict faithfulness to the source material. Addressing this limitation is critical for applications that require highly verifiable responses. Furthermore, we observed a substantial divergence between semantic and syntactic metrics, confirming the limitations of lexical evaluation for generative QA; models frequently produce factually correct answers using entirely different surface forms, which strongly argues for prioritizing semantic evaluation. Additionally, domain adaptation remains a significant hurdle, as performance degrades noticeably when shifting to specialized domains. Finally, our results indicate that model architecture and training methodology may be more decisive factors than raw parameter count, as evidenced by smaller models like Mistral-Nemo (12B) achieving results comparable to much larger models like GPT-4o (¿175B) on several metrics.

4.2.2 QA over Tabular Data

While the preceding sections have focused on question answering over unstructured text, a substantial portion of information in real-world documents—particularly in financial, scientific, and business contexts—is organized in tabular form. Tables present unique challenges for QA systems: they encode information through spatial relationships, headers, and cell alignments that are fundamentally different from the sequential nature of prose text. This section presents our investigation into leveraging large language models for question answering over tables, examining both direct QA approaches and semantic parsing paradigms.

Challenges of QA over Tables

Question answering over tabular data differs fundamentally from text-based QA in several important respects. Tables encode information through a two-dimensional structure where meaning emerges from the intersection of row and column semantics, rather than through sequential narrative. This structural encoding creates unique challenges that must be addressed by any QA system operating on tabular data. Tables exhibit diverse structural patterns including merged cells, hierarchical headers, multi-level column groupings, and nested row categories. Financial reports, in particular, frequently employ complex layouts with grouped columns representing different time periods or business segments, and grouped rows representing hierarchical categories of financial metrics. Interpreting such tables requires understanding not just the cell values but also the structural relationships that determine their meaning. Unlike text-based QA where answers are often extractable spans, table QA frequently requires numerical reasoning—computing sums, differences, percentages, or comparisons across multiple cells. A question such as "What was the year-over-year growth in revenue?" requires identifying the relevant cells, understanding their temporal relationship, and performing arithmetic operations. This reasoning requirement substantially increases task complexity. Tables must be serialized into a format consumable by language models, which fundamentally process sequential text. The choice of serialization strategy—whether plain text, HTML, markdown, or structured representations—significantly impacts model performance. Information about column alignment, header relationships, and cell boundaries may be preserved or lost depending on the representation chosen. Tables in specialized domains such as finance employ conventions and terminology that may differ from general-domain tables. Financial tables often use parentheses to denote negative values, employ specific units (millions, thousands), and contain domain-specific abbreviations. Models must learn to interpret these conventions correctly to provide accurate answers. Traditional approaches to table QA have relied on semantic parsing, where natural language queries are translated into formal query languages such as SQL or logical forms [17, 38]. While effective when sufficient training data is available, these approaches often struggle with cross-domain generalization and require substantial annotation effort. The emergence of large language models offers an alternative paradigm where models can leverage pre-training on diverse data sources to perform both direct question answering and code generation for semantic parsing, potentially providing more flexible solutions for table QA tasks.

Direct QA Approach

The direct QA approach frames table question answering as a straightforward generation task: given a table and a natural language question, the model generates an answer directly without intermediate formal representations. This approach leverages the in-context learning capabilities of large language models, allowing them to interpret tabular data and produce answers in a single inference pass. A critical design decision in direct QA is the choice of table representation format. We investigated three distinct formats to understand their impact on model performance: *Plain Text* represents tables by concatenating cell values with minimal structural markers, removing HTML tags and formatting. While compact, this representation loses explicit information about column boundaries and header relationships, requiring models to infer structure from context.

HTML preserves the original table structure including header tags (`<th>`), row delimiters (`<tr>`), and cell boundaries (`<td>`). This format maintains explicit structural information that mirrors how tables are rendered in web documents, potentially aligning well with patterns seen during model pre-training. *Flattened HTML* addresses the challenge of complex table structures by transforming grouped rows and columns into a normalized form. Using an LLM-assisted preprocessing step, tables with merged cells and hierarchical headers are converted into a flat representation where each row contains complete attribute information. This normalization eliminates structural complexity at the cost of potential information loss and preprocessing overhead. Effective prompting is essential for direct QA performance. Our prompt design incorporates several elements: a clear task description specifying the expected output format (single word or short phrase), explicit instructions regarding units and numerical formatting (e.g., "include units such as $, %, million"), and few-shot examples demonstrating the expected response style. For specialized domains such as financial analysis, prompts are adapted to include domain-specific conventions, such as handling negative values represented with parentheses. We evaluated both proprietary and open-source models to assess the accessibility of high-quality table QA. GPT-4o represents the current state of the art in proprietary models, while Llama 3.1 8B provides an open-source alternative suitable for deployment in resource-constrained or privacy-sensitive environments. This comparison enables assessment of the performance gap between model classes and the trade-offs involved in model selection for production deployment.

Semantic Parsing Approach

As an alternative to direct generation, semantic parsing approaches translate natural language queries into executable code that operates on structured representations of tables. This paradigm offers several potential advantages: generated code is interpretable and auditable, execution errors provide explicit failure signals, and the approach naturally handles complex computations that might challenge direct generation.

We investigated three target languages for semantic parsing:

SQL represents the traditional approach to querying structured data. Tables are converted to relational format with defined schemas, and natural language queries are translated into SQL statements. While SQL is a well-established query language with clear semantics, it may not align optimally with the textual patterns encountered during LLM pre-training.

Pandas-based Python leverages the popular data manipulation library for tabular operations. Tables are represented as DataFrames, and queries are translated into Python functions that use Pandas operations for filtering, aggregation, and computation. This approach benefits from the extensive presence of Pandas code in pre-training corpora.

Vanilla Python generates Python functions that operate on dictionary representations of tables without external library dependencies. This approach produces simpler code that relies only on basic Python operations, potentially reducing the complexity of generated programs and the risk of library-specific errors.

The semantic parsing pipeline operates in multiple stages. First, tables are converted from their original HTML representation into the target data structure (SQL table, Pandas DataFrame, or Python dictionary). Second, an LLM translates the natural language query into executable code in the target language. Third, the generated code is parsed and executed against the data structure, producing a result value. Finally, another LLM call converts the execution result into a natural language response matching the expected answer format.

This multi-stage pipeline introduces additional points of potential failure—code generation errors, type mismatches during execution, and response formatting issues—but provides explicit intermediate outputs that facilitate debugging and error analysis.

Experimentation and Results

We conducted comprehensive experiments to evaluate both direct QA and semantic parsing approaches across multiple datasets and model configurations.

Our evaluation employed four datasets spanning different domains and task types: **FinTabNetQA** comprises 250 question-answer pairs associated with financial tables extracted from annual reports of S&P 500 companies. Questions in this dataset require precise extraction of cell values from complex financial tables, testing models' ability to navigate intricate tabular structures.

VWTQ (Wikipedia Table Questions) contains 750 question-answer pairs associated with Wikipedia tables. Questions range from direct cell extraction to complex reasoning over multiple cells, providing a diverse evaluation across multiple domains.

VTabFact comprises 250 statement-verification pairs where the task is to determine whether a given statement is supported by the data in the associated table, testing models' ability to perform factual verification against tabular evidence.

FinTab-It is a proprietary financial dataset in Italian containing 532 question-answer pairs. This dataset specifically assesses performance on non-English financial documents, addressing the multilingual requirements of practical applications.

Table 4.9 summarizes the characteristics of these datasets.

Table 4.9: Overview of datasets used for table QA evaluation.

Dataset	QA Pairs	Domain	Task	Language
FinTabNetQA	250	Financial	Extraction	English
VWTQ	750	Multiple	Extraction/Reasoning	English
VTabFact	250	Multiple	Verification	English
FinTab-It	532	Financial	Reasoning	Italian

Table 4.10 presents the accuracy of direct QA across models and table representations.

Several important patterns emerge from these results. First, HTML representation consistently outperforms plain text across all datasets and models, with improvements ranging from 3.2 to 20.8 percentage points. This suggests that the structural information encoded in HTML provides crucial guidance for LLMs in interpreting tabular data, likely because HTML table patterns are well-represented in pre-training corpora. Second, GPT-4o demonstrates remarkable robustness

Table 4.10: Accuracy of large language models for direct table QA with different table representations.

Dataset	Model	Plain Text	HTML	Flat HTML
FinTabNetQA	GPT-4o	0.956	0.988	0.928
	Llama 3.1 8B	0.712	0.920	0.856
VTabFact	GPT-4o	0.780	0.856	0.824
	Llama 3.1 8B	0.652	0.696	0.696
VWTQ	GPT-4o	0.607	0.669	0.648
	Llama 3.1 8B	0.355	0.440	0.432
FinTab-It	GPT-4o	0.727	0.725	—
	Llama 3.1 8B	0.372	0.496	—

across formats compared to Llama 3.1 8B. The average performance gap between HTML and plain text representations is only 6.0% for GPT-4o, compared to 15.1% for Llama 3.1 8B. This indicates that smaller models are more reliant on well-structured inputs to compensate for their more limited parametric knowledge. Third, flattened HTML does not provide consistent improvements. While designed to simplify complex table structures, this representation shows worse performance than standard HTML on FinTabNetQA (0.928 vs. 0.988 for GPT-4o) and no meaningful improvement on other datasets. This suggests that grouped columns and hierarchical headers do not represent significant obstacles for current LLMs, and the preprocessing step may introduce information loss that outweighs any structural simplification benefits. Error analysis revealed that Llama 3.1 8B exhibits specific failure patterns including numerical rounding (truncating decimal values) and digit truncation in large numbers. These errors persist despite explicit prompt instructions to preserve exact values, suggesting that few-shot in-context learning or post-generation heuristics may be necessary to mitigate such issues in smaller models. To assess whether supervised fine-tuning can reduce the performance gap between smaller open-source models and larger proprietary alternatives, we conducted experiments using QLoRA [15] to fine-tune Llama 3.1 8B. Table 4.11 presents results for different training configurations. Fine-tuning produces only modest improvements: the best configuration (8 epochs) achieves 0.924 accuracy compared to 0.920 for the baseline, a marginal 0.4% improvement. More concerning, extended training (12 epochs) leads to severe performance degradation (0.480),

Table 4.11: QLoRA fine-tuning configurations and resulting accuracy on FinTabNetQA (HTML tables).

Configuration	Rank	Alpha	Epochs	Accuracy
Baseline (no fine-tuning)	—	—	—	0.920
QLoRA 1	32	16	4	0.912
QLoRA 2	32	16	8	0.924
QLoRA 3	32	16	12	0.480

indicating overfitting on the limited training data. These results suggest that larger and more diverse datasets are required to substantially improve smaller models' table QA capabilities through supervised fine-tuning, and that parameter-efficient methods alone cannot bridge the fundamental capability gap with larger models. Table 4.12 presents the performance of GPT-4o using different semantic parsing paradigms on FinTabNetQA.

Table 4.12: Performance of GPT-4o for table QA with semantic parsing on FinTabNetQA.

Metric	SQL	Pandas	Vanilla Python
Accuracy	0.356	0.788	0.860
Execution Errors	2	10	7

Several findings merit discussion. First, vanilla Python substantially outperforms both SQL and Pandas, achieving 0.860 accuracy compared to 0.788 for Pandas and only 0.356 for SQL. This suggests that LLMs are more proficient at generating standard Python code than specialized query languages or library-specific APIs, likely reflecting the distribution of code in pre-training data.

Second, semantic parsing underperforms direct QA on HTML tables (0.860 vs. 0.988 for GPT-4o). This counterintuitive result suggests that for well-structured tables, the additional complexity of the semantic parsing pipeline—including code generation, execution, and result interpretation—introduces more error opportunities than it resolves. However, semantic parsing offers advantages in interpretability and auditability that may justify its use in high-stakes applications where understanding the reasoning process is essential. Third, execution errors reveal systematic issues in code generation. SQL queries frequently fail due to type

casting errors when columns contain string values with embedded formatting (e.g., currency symbols), as the generated code assumes numeric types. Pandas code exhibits similar type inference issues. Vanilla Python errors most commonly occur when the model generates example usage code alongside the function, causing parsing failures. These error patterns suggest that providing schema information or example rows in the prompt could improve code generation reliability. Our investigation into table QA yields several actionable insights:

1. **Table representation matters significantly.** HTML format should be preferred over plain text for table QA, providing improvements of 8–20% depending on model and dataset. This structural information is particularly critical for smaller models.

2. **Model size influences format sensitivity.** Larger models (GPT-4o) demonstrate robust performance across formats, while smaller models (Llama 3.1 8B) show substantial degradation with less structured inputs. This suggests that resource-constrained deployments should prioritize input quality.

3. **Direct QA outperforms semantic parsing for well-structured tables.** When tables are provided in HTML format, direct generation achieves higher accuracy than semantic parsing approaches, suggesting that the interpretability benefits of semantic parsing come at a performance cost.

4. **Vanilla Python is the optimal target language for semantic parsing.** When semantic parsing is required—for interpretability, auditability, or handling complex computations—generating vanilla Python code yields better results than SQL or Pandas-based approaches.

5. **Limited data constrains fine-tuning effectiveness.** Parameter-efficient fine-tuning on small datasets produces marginal improvements and risks overfitting, indicating that substantial training data is required to meaningfully improve smaller models' table QA capabilities.

These findings have direct implications for deploying table QA systems in practical applications. For maximum accuracy with minimal engineering effort, direct QA with HTML representation using capable models such as GPT-4o is recommended. For applications requiring interpretable reasoning traces or operating under cost constraints, semantic parsing with vanilla Python generation provides a viable alternative with acceptable performance trade-offs.

Chapter 5

Retrieval Augmented Generation Systems Optimization

5.1 State of the Art

Retrieval-Augmented Generation (RAG) has emerged as a pivotal paradigm in modern natural language processing, addressing fundamental limitations of large language models by combining neural retrieval mechanisms with generative capabilities. This section provides a comprehensive overview of RAG architectures, their variants and optimizations, and the critical role of document chunking strategies in determining system effectiveness.

5.1.1 Foundational RAG architecture

The foundational RAG architecture was introduced by Lewis et al. [42], who proposed a novel approach to knowledge-intensive NLP tasks by combining a pre-trained neural retriever with a pre-trained sequence-to-sequence generator. This seminal work demonstrated that augmenting language models with external knowledge through retrieval could significantly improve performance on tasks requiring factual accuracy while reducing the phenomenon of hallucination—the generation of plausible but factually incorrect content [78].

Formally, RAG models the probability of generating a target sequence y given an input query x by treating the retrieved documents (or passages) z as a latent variable. The model retrieves a set of documents z from a knowledge base using a retriever with parameters η, and generates the output using a generator with parameters θ. The probability of the generated sequence is obtained by marginalizing over the latent documents:

$$P_{\text{RAG}}(y|x) = \sum_{z \in Top\text{-}K(x)} P_\eta(z|x) \prod_{i=1}^{N} P_\theta(y_i|x, z, y_{1:i-1}) \tag{5.1}$$

In this formulation, $P_\eta(z|x)$ represents the retrieval probability (usually based on dense vector similarity), and $P_\theta(y_i|\dots)$ represents the generation probability of the next token given the context and the retrieved document. The objective is to minimize the negative log-likelihood of the target outputs $\mathcal{L} = -\log P_{\text{RAG}}(y|x)$.

Prior to RAG, related approaches such as REALM [26] demonstrated the viability of retrieval-augmented pre-training, where a knowledge retriever was jointly trained with the language model. However, RAG's post-hoc retrieval approach proved more practical for deployment, as it decouples the retrieval and generation components, enabling greater flexibility in system design and knowledge base management.

The standard RAG pipeline comprises four interconnected phases that operate in sequence to transform user queries into accurate, grounded responses [59]. The first phase, *ingestion*, processes input documents to create manageable and searchable units. Documents are segmented into smaller parts, commonly referred to as "chunks," using various strategies ranging from fixed-size splitting to semantically-aware segmentation. For visually-oriented documents such as PDFs, Document Layout Analysis techniques can be exploited to recognize more meaningful document boundaries. These chunks are then converted into dense vector representations through embedding models, transforming textual information into high-dimensional vectors that capture semantic essence. The resulting embeddings are ingested into vector stores optimized for efficient similarity search operations, such as Milvus, Pinecone or Weaviate.

Upon receiving a user query, the *retrieval* phase encodes it using the same embedding model employed during ingestion. The query vector then undergoes similarity search against the indexed document embeddings to identify the k most relevant chunks. This phase narrows down the potentially vast information space to the most pertinent passages for answer generation. The choice of retrieval method—whether dense, sparse, or hybrid—significantly impacts both the relevance of

retrieved content and system latency.

In the *generation* phase, a Large Language Model synthesizes information from the retrieved chunks to construct a coherent and contextually appropriate response. The generation phase conditions the LLM on both the original query and the retrieved context, enabling the model to produce answers that are grounded in the source material rather than relying solely on parametric knowledge. This grounding mechanism is crucial for reducing hallucinations and ensuring factual accuracy.

Finally, the *evaluation* phase employs both ground-truth dependent and independent metrics. Ground-truth dependent metrics assess correctness against predefined reference answers, while ground-truth independent metrics such as context relevance, groundedness, and answer relevance [19] evaluate system performance without requiring gold-standard responses. This dual approach enables thorough assessment of retrieval quality, generation faithfulness, and overall system utility.

The RAG paradigm offers several advantages over pure generative approaches. First, it enables access to knowledge beyond the model's training cutoff, allowing systems to incorporate current information. Second, retrieved passages provide explicit evidence for generated answers, enhancing interpretability and enabling source attribution. Third, the modular architecture allows independent updates to the knowledge base without requiring model retraining, significantly reducing maintenance costs in production environments.

5.1.2 Advanced RAG variants

The success of the foundational RAG architecture has spawned numerous variants and optimizations designed to address specific limitations and enhance performance across different dimensions. These innovations span query processing, retrieval mechanisms, and the integration between retrieval and generation components. Query expansion techniques leveraging large language models have emerged as powerful approaches for improving retrieval without modifying the underlying index. HyDE (Hypothetical Document Embeddings) [23] introduced a paradigm shift by using LLMs to generate hypothetical documents before embedding: rather than directly embedding a query, HyDE instructs an LLM to generate what a relevant answer document might look like, then embeds that synthetic document for similarity search. This approach achieves performance comparable to supervised dense retrievers without any labeled training data, effectively bridging the semantic gap between short queries and longer documents.

Query2Doc [87] takes a related approach by concatenating LLM-generated pseudo-documents with the original query text, demonstrating improvements of 3-15% on benchmark datasets while benefiting both sparse and dense retrievers. The key practical trade-off for both methods is latency overhead, as LLM inference at query time adds significant processing time that must be balanced against retrieval quality improvements. Hybrid retrieval methods have gained prominence by combining lexical and semantic approaches to capture complementary aspects of relevance [11]. These approaches typically integrate BM25 or other sparse retrievers with dense neural models, addressing limitations of pure dense retrieval including the inability to perform exact keyword matching, sensitivity to out-of-vocabulary terms, and potential failures on queries requiring precise lexical matching. Rank fusion techniques provide mechanisms for combining results from multiple retrieval systems. Reciprocal Rank Fusion (RRF) [14] computes a combined score for each document based on its rank in each constituent system without requiring score normalization. However, recent research has shown that alternatives can improve performance: Convex Combination outperforms RRF in both in-domain and out-of-domain settings, while score-aware hybrid retrieval using weighted combination of dense and sparse scores demonstrates average improvements of 4-8% over RRF on BEIR benchmarks. Reranking has become an essential component of modern retrieval pipelines, operating as a second stage that refines initial results to improve precision. The standard two-stage pattern retrieves 50-100 candidates with fast first-stage methods, then reranks to select the top 5-10 documents before generation, dramatically reducing computational cost while maintaining high answer quality. Cross-encoder rerankers represent the most established approach, employing full attention over concatenated query-document pairs to compute relevance scores [56]. Beyond retrieval enhancements, several architectural innovations address limitations of the standard RAG pipeline. Self-RAG [4] introduces reflection tokens that enable the model to adaptively retrieve information and critique its own generations, allowing the system to decide when retrieval is necessary, assess the relevance of retrieved passages, and evaluate the quality of generated responses. CRAG (Corrective RAG) [96] addresses the challenge of low-quality retrieval by implementing a lightweight retrieval evaluator that assesses relevance confidence and triggers corrective actions—such as refined queries or web search—when initial retrieval fails to produce satisfactory results. RAFT (Retrieval Augmented Fine-Tuning) [100] adapts language models to domain-specific RAG scenarios through targeted fine-tuning that teaches models to distinguish between relevant and irrelevant retrieved content, improving robustness to retrieval noise. FunnelRAG [93] proposes a coarse-to-fine progress-

ive retrieval paradigm that balances retrieval effectiveness with computational efficiency through hierarchical candidate filtering. A critical challenge in RAG systems is the "Lost in the Middle" phenomenon identified by Liu et al. [48], where models struggle to maintain attention across long input contexts. This limitation is particularly relevant for QA systems processing extensive documents or integrating information from multiple retrieved passages. Research has shown that information placed in the middle of long contexts receives less attention than information at the beginning or end, motivating strategies such as careful passage ordering and context compression. The emergence of long-context language models with extended context windows (100K+ tokens) presents both opportunities and challenges for RAG systems. While these models can theoretically process entire documents without chunking, empirical evaluations reveal that RAG approaches often maintain advantages in terms of answer accuracy, source attribution capability, and computational efficiency for many practical applications.

5.1.3 Source Attribution

Source attribution—linking generated text to its underlying sources—is critical for the transparency, trustworthiness, and verifiability of RAG system outputs [5]. By connecting generated content to portions of the retrieved context, source attribution supports fact-checking, grounded summarization, and verifiable question answering. It enables users to assess the quality and relevance of the sources behind each claim, making it a key component for responsible AI deployment. Two related but distinct tasks address the challenge of connecting generated content to sources. Source attribution involves identifying supporting materials for generated content, either during or after generation, and does not necessarily require a RAG pipeline. Context attribution, more specifically relevant to RAG systems, links LLM-generated sentences with parts of the retrieved input context [99]. Both tasks can be implemented using in-line citations, which connect each sentence to its supporting source, or answer-level attribution, which provides sources collectively at the end of the output. Early approaches to source attribution were predominantly LLM-based and designed for open-domain question answering. WebGPT [55] relies on a text-based web browsing environment and an LLM to trigger information extraction from web pages, though it focuses primarily on during-generation in-line citations. RARR [22] uses search engines to identify evidence supporting generated answers, employing retrieved evidence for answer revision and attribution. More recent approaches have explored training-based methods: some works train models with reinforcement learning to generate answers with quotes,

while others employ factual consistency models to filter synthetically generated data and use focused learning to concentrate backpropagated information around generated answers [2]. LongCite [99] uses a retriever approach to identify citations for synthetically generated answers combined with focused learning for tuning. However, these approaches typically rely on expensive LLM fine-tuning strategies. Among zero-shot approaches, several works have explored post-hoc and RAG-based methods for generating text with citations. Some approaches use source attribution with in-context learning after open-book generation for answer revision, while others employ fine-tuning to train LLMs to generate answers with citations or to function as classifiers identifying citations in outputs generated by any LLM [52].

5.2 Contributions

5.2.1 Continuous Improvement

A fundamental challenge in deploying RAG systems in production environments is the need for continuous adaptation and improvement as the system interacts with real users across diverse business domains. While initial model selection and hyperparameter tuning provide a solid foundation, the true value of a production RAG system emerges through its ability to learn from operational feedback and systematically improve each component of the pipeline. This section presents our methodology for iterative enhancement of RAG systems, centered around a novel data structure called the **RAG-Dataset**, which serves as the foundation for collecting, organizing, and leveraging user feedback to improve all pipeline components.

RAG-Dataset

Standard information retrieval benchmarks such as BEIR [81] provide valuable resources for evaluating retrieval performance in zero-shot settings. The BEIR benchmark format captures the essential elements of a retrieval task: a corpus of documents, a set of queries, and query-relevance annotations (qrels) that map queries to their relevant documents with graded relevance scores. However, this format was designed primarily for retrieval evaluation and lacks the necessary structure to support end-to-end RAG system optimization, particularly for the generation and source attribution components.

To address this limitation, we introduce the **RAG-Dataset** format, an extension of the BEIR schema that augments the standard retrieval-focused structure with additional metadata required for comprehensive RAG system improvement. The RAG-Dataset maintains backward compatibility with BEIR, ensuring that existing evaluation tools and workflows remain applicable, while introducing several key extensions. The RAG-Dataset is organized as a collection of JSON files, each serving a specific purpose in the evaluation and improvement pipeline. The complete structure comprises the following components:

1. **Corpus**: Contains the document collection segmented into retrievable chunks. Each chunk entry includes:

   - `page_content`: The textual content of the chunk
   - `metadata`: A structured object containing `id` (unique chunk identifier), `document_id`, `document_name`, `page` number, `start_index`, `end_index`, and `type`

   The chunk identifier follows a hierarchical naming convention: {`document_id`}_{`page`}_{`start_index`}_{`end_index`}, enabling precise localization within the source document.

2. **Queries**: A dictionary mapping query identifiers to natural language questions. Query identifiers follow the pattern {`session_id`}_`question`, linking each query to its originating user session.

3. **Answers**: A dictionary mapping answer identifiers to the generated response text. Answer identifiers encode both the originating query and the answer source: {`query_id`}_`answer`_{`source_type`}, where `source_type` indicates the provenance:

   - `system_answer` for system-generated answers.
   - `user_answer` for user-provided ground truth.
   - `expert_answer` for expert-annotated ground truth.

A distinctive feature of the RAG-Dataset is its support for relevance annotations at multiple granularities, enabling fine-grained analysis of retrieval performance:

1. **Chunk-level Qrels** (`chunk_qrels.json`): Maps each query to its relevant chunks with associated feedback metadata. Each entry contains:

- `score`: Binary relevance indicator ($+1$ for relevant, -1 for irrelevant)
- `answer_feedback`: User feedback on the answer quality (`positive`/`negative`)
- `reference_feedback`: User feedback on the passage relevance (`positive`/`negative`)

2. **Document-level Qrels** (`document_qrels.json`): Aggregates relevance at the document level, using the same metadata structure. This dual-level annotation supports both fine-grained chunk retrieval optimization and coarse-grained document-level analysis.

3. **Answer Qrels** (`answer_qrels.json`): Maps queries to their associated answers with quality feedback. Each entry includes:

   - `score`: Answer quality indicator
   - `answer_feedback`: User assessment of answer correctness
   - `dataset_type`: Source classification (`altilia_gpt`, `user_answer`, `expert_answer`)

This multi-level structure enables independent optimization of retrieval (using chunk and document qrels) and generation (using answer qrels) components, while the cross-referenced feedback signals support source attribution model training. The RAG-Dataset includes a comprehensive statistics file (`stats.json`) that provides aggregate metrics for corpus analysis and quality monitoring:

- **Corpus statistics**: Number of documents and chunks, with distributional metrics (average, min, max, median, standard deviation) for page count, chunk count, word count, and character count per document and chunk
- **Query statistics**: Total query count, chunks per query distribution, and documents per query distribution

These statistics enable automated monitoring of dataset growth and quality, supporting decisions about when to trigger model retraining based on data accumulation thresholds.

Feedback Collection Interface

The practical implementation of continuous improvement relies on an effective mechanism for collecting user feedback without disrupting the user experience.

Our system integrates feedback collection directly into the response interface, presenting users with intuitive controls for providing signals on both retrieved passages and generated answers. Following generation, the system displays not only the answer but also the set of passages that the language model utilized in constructing its response. This transparency serves dual purposes: it enables users to verify the factual basis of the answer (supporting trust and accountability), and it provides the interface for collecting passage-level relevance feedback. Each displayed passage includes simple binary feedback controls (represented visually as "thumbs up" and "thumbs down" icons) that allow users to quickly indicate relevance. The RAG-Dataset captures two distinct feedback dimensions for each retrieved passage:

- **Reference feedback** (`reference_feedback`): Indicates whether the passage is genuinely relevant to answering the query, regardless of the answer quality. This signal directly informs retrieval component optimization.
- **Answer feedback** (`answer_feedback`): Indicates whether the generated answer adequately addressed the user's information need. This signal drives generation model improvement.

The separation of these feedback dimensions is critical for diagnosing system failures. For instance, a query with positive reference feedback but negative answer feedback indicates that retrieval succeeded but generation failed—the correct passages were found, but the LLM failed to synthesize an appropriate response. Conversely, negative reference feedback with positive answer feedback (a rarer case) suggests that the model generated a satisfactory response despite receiving suboptimal context, potentially through reliance on parametric knowledge. Each annotated example is classified into one of several categories based on feedback signals and provenance:

- `positive`: Examples with positive reference feedback, used as positive training examples
- `negative`: Examples with negative reference feedback, used as hard negatives
- `user_passages`: User-provided relevant passages
- `user_answer`: User-provided reference answers
- `expert_answer`: Expert-annotated reference answers

- `system`: System-generated answers for evaluation

This classification enables stratified sampling and weighted training strategies that prioritize higher-quality annotations while still leveraging the larger volume of system-generated examples.

Iterative Component Improvement

The RAG-Dataset serves as the foundation for systematic improvement of each component in the RAG pipeline. Our architecture decomposes the system into four primary trainable components, each benefiting from different aspects of the collected feedback: The bi-encoder embedding model, which generates dense vector representations for both queries and passages, is fine-tuned using a combination of synthetic data (as described in Chapter 3) and real user feedback from the RAG-Dataset. The synthetic data generation approach, based on contrastive learning with Multiple Negative Ranking (MNR) loss, provides a strong initialization, while user feedback enables domain-specific refinement. Training data is constructed from the chunk-level qrels as follows:

- **Positive pairs**: Queries paired with chunks having `score: 1` and `reference_feedback: positive`
- **Hard negatives**: Chunks with `score: -1` that were retrieved but received negative reference feedback. These are particularly informative as they represent failure cases where the current model incorrectly ranked irrelevant passages highly.

The cross-encoder reranker, which refines initial retrieval results by jointly encoding query-passage pairs, benefits from the dual-level qrels structure. Document-level qrels provide coarse-grained supervision for ranking documents, while chunk-level qrels enable fine-grained passage ranking optimization. The reranker is trained when sufficient feedback accumulates to indicate suboptimal ranking behavior. The generative language model that synthesizes answers from retrieved context is adapted using feedback from the answer qrels. This adaptation employs parameter-efficient fine-tuning techniques, specifically LoRA (Low-Rank Adaptation) [30], to update the model's behavior while preserving its general capabilities. Training data is constructed by pairing queries with their retrieved context and answers:

- **Positive examples**: Query-context-answer triplets where `answer_feedback` is `positive`

- **Reference examples**: Entries with `dataset_type` of `user_answer` or `expert_answer` provide the highest-quality supervision
- **Contrastive pairs**: When both positive and negative answers exist for the same query, preference-based training can be applied

The distinction between `altilia_gpt` (system-generated), `user_answer` (user-provided), and `expert_answer` (expert-annotated) answers enables curriculum learning strategies that progressively incorporate higher-quality supervision. The source attribution component identifies which retrieved passages genuinely support claims in the generated answer. The RAG-Dataset's separation of `answer_feedback` and `reference_feedback` provides ideal supervision for this task. Training examples are derived from entries where these signals diverge:

- **Positive attribution**: Passages with positive reference feedback for queries with positive answer feedback (the passage supported a correct answer)
- **Negative attribution**: Passages with negative reference feedback despite positive answer feedback (the passage was not actually used to generate the correct answer)

A critical design principle of our approach is the creation of separate RAG-Dataset instances for each business domain or application context. Rather than attempting to build a single universal improvement dataset, we recognize that optimal system behavior varies significantly across domains. A legal document retrieval system requires different optimization signals than a customer support knowledge base or a technical documentation search system.

Each domain-specific RAG-Dataset accumulates feedback from users operating within that domain, enabling targeted optimization that preserves the unique characteristics and requirements of each application context. The statistics file tracks domain-specific distributional properties:

Listing 5.1: Example statistics for a banking domain RAG-Dataset

```
{
  "corpus": {
    "Document": {
      "#": 2419,
      "num_pages": {"average": 16.97, "median": 7, "max":
          977},
```

```
      "num_chunks": {"average": 17.54, "median": 7, "max":
          1099}
    },
    "Chunk": {
      "#": 42432,
      "num_words": {"average": 281.56, "median": 285.0},
      "num_chars": {"average": 2110.97, "max": 4096}
    }
  },
  "queries": {
    "#": 308,
    "chunks_per_query": {"average": 5.33, "median": 5.0},
    "docs_per_query": {"average": 2.81, "median": 3.0}
  }
}
```


These statistics inform chunking strategy optimization (the 4096 character maximum suggests a deliberate choice balancing context length with retrieval granularity) and retrieval depth configuration (the average of 5.33 chunks per query indicates typical multi-passage reasoning requirements). To operationalize the continuous improvement methodology, we implement an automated pipeline that orchestrates feedback collection, data preparation, model training, and deployment. This pipeline enables RAG systems to improve autonomously as they accumulate operational experience, while maintaining appropriate human oversight. User feedback is continuously collected and aggregated into the domain-specific RAG-Dataset. The dual feedback structure (`answer_feedback` and `reference_feedback`) enables nuanced quality filtering:

- Examples with consistent positive signals across both dimensions are prioritized for positive training
- Examples with divergent signals (e.g., positive answer but negative reference) are flagged for potential model diagnosis
- Examples with consistent negative signals inform hard negative mining

Model retraining is triggered based on configurable conditions monitored through the statistics file:

- **Volume threshold**: Sufficient new feedback has accumulated (e.g., query count exceeds previous training snapshot by a configurable margin)

- **Distribution shift**: Significant changes in chunks-per-query or docs-per-query distributions indicate evolving user behavior
- **Feedback ratio changes**: Shifts in the proportion of positive vs. negative feedback suggest model degradation
- **Scheduled intervals**: Regular retraining on a fixed schedule regardless of other triggers

The continuous improvement pipeline integrates with CI/CD infrastructure to manage model versioning, deployment, and rollback capabilities. Each model version is tagged with the RAG-Dataset snapshot (identified by statistics and qrels file hashes) used for training, enabling reproducibility and debugging. The `dataset_type` field supports A/B testing by allowing different model versions to be evaluated against held-out user provided examples.

Benefits and Industrial Impact

The RAG-Dataset methodology and associated continuous improvement pipeline provide several significant benefits in industrial deployments: By leveraging real user feedback with structured dual-signal annotation, the system achieves domain-specific optimization without requiring expensive manual annotation campaigns. The organic collection of feedback through normal system operation amortizes the data acquisition cost across productive use of the system. The separation of answer feedback and reference feedback enables precise diagnosis of system failures. Operations teams can identify whether issues stem from retrieval (wrong passages), generation (wrong answer synthesis), or source attribution (incorrect citation), and prioritize improvement efforts accordingly. The dual-level qrels structure (chunk and document) supports optimization at multiple retrieval granularities, enabling systems to balance precision (chunk-level) and recall (document-level) based on application requirements. The structured `dataset_type` classification maintains clear provenance for all training data, supporting audit requirements and enabling quality-weighted training strategies that prioritize human-validated examples.

Through the RAG-Dataset framework and automated improvement pipeline, we transform RAG systems from static deployments into continuously learning systems that improve with every user interaction. This approach has been successfully deployed across multiple business domains, demonstrating consistent improvement in retrieval quality, answer accuracy, and user satisfaction over extended operational periods.

5.2.2 Optimal Chunking for RAG with Long Context Models

The rapid expansion of context windows in Large Language Models—from the original 512 tokens of BERT to the millions of tokens supported by recent architectures such as Gemini and Claude—has fundamentally altered the landscape of Retrieval-Augmented Generation. These long-context capabilities raise a critical question for RAG system designers: when models can potentially ingest entire document collections in a single forward pass, is the traditional retrieve-then-generate paradigm still necessary? Despite the remarkable expansion of context windows, Retrieval-Augmented Generation maintains several compelling advantages that ensure its continued relevance in production systems. Understanding these advantages is essential for making informed architectural decisions. The computational cost of transformer-based models scales quadratically with sequence length due to the self-attention mechanism. While various optimizations such as flash attention and sliding window approaches have reduced this burden, processing extensive contexts remains substantially more expensive than targeted retrieval. In production environments where cost-per-query directly impacts business viability, RAG systems that retrieve only relevant passages can achieve comparable accuracy at a fraction of the computational cost. Our experiments demonstrate that strategic chunking and retrieval can reduce input token counts by 80–95% compared to whole-document approaches while maintaining or even improving answer quality in specific domains. As documented by Liu et al. [48], language models exhibit systematic biases in how they distribute attention across long contexts. Information placed in the middle of extensive inputs receives disproportionately less attention than content at the beginning or end—a phenomenon that persists even in models explicitly designed for long-context processing. RAG systems mitigate this limitation by surfacing relevant passages to prominent positions in the context, ensuring that critical information receives appropriate attention during generation. This architectural advantage becomes increasingly important as the ratio of relevant to irrelevant content decreases in larger contexts. In knowledge-intensive applications such as legal analysis, medical information retrieval, and financial compliance, the ability to trace generated claims to specific source passages is not merely desirable but often legally required. RAG architectures naturally support fine-grained source attribution by maintaining explicit links between retrieved passages and generated content. Whole-document approaches, while capable of source attribution in principle, face practical challenges in identifying which specific portions of extensive documents support particular claims. RAG systems enable knowledge updates without model retraining by modifying the external

document collection. This capability is particularly valuable in domains where information changes rapidly or where maintaining multiple knowledge bases for different applications is required. Long-context approaches that rely on in-context learning must re-process entire document collections for each query, whereas RAG systems can incrementally update their knowledge bases through efficient index modifications. Recent empirical studies provide nuanced perspectives on the RAG versus long-context tradeoff. Li et al. [46] found that while long-context models can outperform RAG in average performance when sufficiently resourced, RAG maintains significant advantages in cost-efficiency. Conversely, Xu et al. [95] demonstrated that a 4K-context LLM with retrieval could match a 16K finetuned LLM, suggesting that retrieval consistently benefits LLMs regardless of their native context window size. The LaRA benchmark [43] revealed that the optimal choice depends on multiple factors including model size, task type, and context length, with RAG often proving more beneficial for weaker models or as context length increases. Li et al. [44] also found that long-context models generally perform better, especially for Wikipedia-based QA, but RAG excels in dialogue-based queries. These findings underscore that the choice between RAG and long-context approaches is not binary but rather a design decision that should be informed by specific application requirements. The choice of document segmentation strategy fundamentally shapes RAG system performance by determining the granularity at which information can be retrieved and the coherence of context provided to the generation model. We systematically evaluate three distinct approaches representing different points on the granularity spectrum.

Whole Document Retrieval. In this strategy, documents are provided to the retrieval system and subsequently to the LLM in their entirety without prior segmentation. This approach directly tests the long-context processing capabilities of modern LLMs and their resilience to potentially noisy or extensive information within a single block. While whole-document retrieval preserves complete original context and eliminates the risk of fragmenting semantically connected content, it presents challenges in terms of processing efficiency and the potential dilution of relevant information within vast amounts of text. This strategy is particularly relevant for documents with high internal coherence where cross-references and long-range dependencies are prevalent.

Naive Fixed-Size Chunking. Fixed-size chunking segments documents into smaller pieces using a straightforward approach: text is initially split sentence

by sentence, and these sentences are then aggregated into chunks adhering to a predefined token limit. This ensures a degree of uniformity in chunk size across the corpus, which can be beneficial for batch processing and model input consistency. The primary advantage of this approach lies in its simplicity and reproducibility—identical parameters yield identical segmentations regardless of document content or structure.

However, the content-agnostic nature of fixed-size chunking introduces significant limitations. The method may inadvertently sever semantic connections by splitting paragraphs or even complex sentences across chunk boundaries when the token limit is reached. Studies exploring optimal chunk sizes have suggested that 256 or 512 tokens often provide reasonable tradeoffs between faithfulness and relevancy [88], though optimal values vary considerably across domains and tasks.

Semantic Chunking. Semantic chunking aims to overcome the limitations of naive approaches by leveraging the inherent structure of documents. Rather than imposing arbitrary boundaries, this strategy divides text based on natural sections—chapters, main sections, subsections, or distinct thematic paragraphs identified by structural cues. The primary goal is to create chunks that maintain high internal contextual coherence, preserving semantically meaningful units that can stand alone as informative passages.

The effectiveness of semantic chunking depends critically on the quality and consistency of source document formatting. Well-structured documents with clear hierarchical organization benefit substantially from this approach, while poorly formatted or inconsistently structured documents may yield suboptimal segmentations. Recent work by Zhao et al. [102] argues that traditional semantic chunking can be inadequate for subtle contextual nuances, proposing LLM-based chunking methods that produce more distinct and cohesive chunks according to metrics such as Boundary Clarity and Chunk Stickiness. However, Qu et al. [66] demonstrated that semantic chunking does not guarantee consistently better performance than simpler fixed-size approaches and that the computational cost of sophisticated segmentation methods is not always justified.

Experimental Methodology

Our experimental framework is designed to isolate the impact of chunking strategy as the primary variable while maintaining consistent, state-of-the-art components for all other pipeline elements. To ensure efficient and relevant context retrieval,

we employed a sophisticated hybrid search pipeline combining the strengths of different retrieval paradigms:

- **Keyword-based Retrieval:** We employed BM25 (Best Matching 25), ranking documents based on term frequency, inverse document frequency, and document length normalization. This component excels at identifying chunks containing exact keyword matches to the user's query and provides a robust baseline for lexical relevance.
- **Semantic Retrieval:** Complementing keyword-based search, we integrated semantic retrieval using multilingual-e5-large embeddings with cosine similarity. This model captures nuanced semantic relationships and supports multilingual content through specialized prefixes during embedding. Critically, we fine-tuned this model on domain-specific data to improve generalization toward the legal domain, following the methodology presented in Section 3.2.2.
- **Reranking:** To refine initial retrieval results, we incorporated gte-multilingual-reranker-base. This cross-encoder takes the top N candidates from preceding stages and re-evaluates their relevance using a more computationally intensive but accurate model, pushing the most relevant chunks to the forefront before they are passed to the LLM.

The final stage of our RAG pipeline presents retrieved and reranked context to advanced LLMs for answer generation. We selected three models representing diverse architectural approaches and capability levels:

- **GPT-4.1**: OpenAI's flagship model, representing the state of the art in proprietary LLMs with extensive long-context capabilities.
- **Gemini 2.0 Flash**: Google's efficient long-context model, optimized for speed while maintaining strong performance on knowledge-intensive tasks.
- **Llama 4 Maverick**: Meta's open-weight model, demonstrating competitive performance with full architectural transparency and fine-tuning capability.

These models are tasked with synthesizing coherent and accurate answers based solely on the contextual information provided by the retrieval system, enabling direct assessment of how different chunking strategies influence generation quality. These specific models were selected to evaluate the chunking strategies across different architectural paradigms and context-window optimizations. GPT-4.1 and

Gemini 2.0 Flash represent frontier proprietary models with highly optimized long-context processing capabilities, while Llama 4 Maverick provides a state-of-the-art open-weight baseline, ensuring our findings regarding document segmentation are robust across both closed and open ecosystems. We utilized two distinct datasets to evaluate chunking strategy robustness across different domains and document structures.

- **CovidQA (Scientific Domain).** This public dataset serves as our benchmark for scientific domain performance, comprising 124 curated question-answer pairs derived from 83 scientific papers regarding COVID-19. The documents are characterized by standard scientific structuring with moderate length: average tokens per document of 3,847, maximum of 8,355, and average tokens per section of 369. This dataset represents scenarios where relevant information is embedded within moderately sized documents of highly technical content.

- **BankReg-IT (Legal Domain).** To test performance in a high-complexity legal environment, we utilized a proprietary dataset composed of Italian banking regulations and normative documents. This dataset presents significantly greater challenges: 572 questions (filtered via BEIR cutoff), average tokens per document of 28,109 (approximately 7.3$\times$ longer than CovidQA), and maximum tokens per document of 265,722. The sheer volume of text per document serves as a stress test for long-context capabilities versus the precision of retrieval systems.

The contrast between these datasets—short, structured scientific texts versus long, complex legal documents—enables assessment of how domain characteristics influence optimal chunking strategies. Given the complexity of legal documents, we applied aggressive cleaning heuristics to normalize text for semantic chunking. This process involved converting text to lowercase, normalizing Unicode characters, removing non-alphanumeric characters, removing explicit section numbering to focus on semantic content, and standardizing whitespace. This preprocessing yielded approximately 9,243 sections across 187 documents, with a mismatch rate of roughly 2% in section indexing—deemed acceptable for the experimental scale.

Experimental Results

We evaluated model performance across different granularity levels, with results categorized by dataset to illuminate domain-specific patterns. To rigorously

quantify generation quality without relying solely on expensive human annotation, we adopted an LLM-as-a-judge evaluation framework. Specifically, we employed GPT-4.1 as the automated evaluator to assess the generated responses. The evaluator was tasked with measuring the factual equivalence between the generated answer and the ground truth, ultimately computing an aggregate answer quality metric on a 0-1 scale. Table 5.1 presents performance on the long-context banking regulation dataset, exploring granularity ranging from whole documents to fine-grained text sections. Several important patterns emerge from the legal domain

Table 5.1: Performance on BankReg-IT (Proprietary Legal Dataset). Metrics indicate aggregate answer quality on a 0–1 scale.

Model	Whole Docs	Naive Chunks	Pages	Sections (Low)	Sections (High)
GPT-4.1	0.672	0.684	0.678	0.600	0.660
Gemini 2.0 Flash	0.642	0.642	0.672	0.563	0.624
Llama 4 Maverick	**0.721**	0.654	0.672	0.587	0.654

evaluation: Llama 4 achieved the highest overall score (0.721) using the whole document strategy, substantially outperforming all other model-strategy combinations. This result suggests that for complex legal reasoning, having the full context without fragmentation allows the model to better resolve inter-dependencies within regulatory texts. For both Llama 4 and GPT-4.1, performance generally degraded as chunking became more granular. Lower-level sections performed worst at approximately 0.60, representing a significant drop from whole-document approaches. This pattern indicates that legal documents possess characteristics that make them particularly sensitive to context fragmentation. While whole document retrieval yielded the best metrics for Llama 4, it was also the most computationally expensive strategy. Organizations must weigh the accuracy benefits against increased inference costs, particularly for high-volume applications. Table 5.2 presents results for the public scientific dataset, which features significantly shorter document lengths and more localized information distribution.

The scientific domain reveals strikingly different patterns: Unlike the legal dataset, the higher-order sections strategy outperformed or matched whole documents for all models. Llama 4 Maverick achieved its peak performance (0.717) using higher-order sections—substantially better than its whole-document performance (0.701). Given that CovidQA documents average only 3,800 tokens, the computational penalty for ingesting whole documents is relatively modest. Nevertheless, semantic chunking provided measurable advantages in precision,

Table 5.2: Performance on CovidQA (Public Scientific Dataset).

Model	Whole Docs	Naive Chunks	Higher-Order Sections
GPT-4.1	0.669	0.661	0.677
Gemini 2.0 Flash	0.677	0.685	0.685
Llama 4 Maverick	0.701	0.629	**0.717**

likely by reducing noise from irrelevant sections within scientific papers. mAll three models showed more consistent performance across chunking strategies in the scientific domain compared to the legal domain, suggesting that shorter, better-structured documents are more forgiving of suboptimal segmentation choices. The dichotomy in results between BankReg-IT and CovidQA demonstrates conclusively that no single chunking strategy provides universally optimal performance for RAG systems. The underperformance of granular chunking strategies in the legal domain reveals fundamental characteristics of regulatory documents that distinguish them from other text types. Legal documents exhibit what we term "high semantic density"—a property where meaning is distributed across extensive spans of text with frequent cross-references and definitional dependencies. A regulation in Section 1 may explicitly reference definitions established in Section 10, creating long-range dependencies that are severed by any form of document fragmentation. Furthermore, legal reasoning often requires understanding the hierarchical relationship between general principles and specific provisions. When documents are chunked, even semantically, this hierarchical context is lost, forcing the LLM to reason with incomplete information. Llama 4 Maverick's ability to effectively handle whole documents averaging 28,000 tokens demonstrates that for high-stakes domains with complex internal structure, the long-context approach—despite higher inference costs—delivers superior accuracy. Conversely, scientific literature exhibits a different information architecture. Research papers are typically structured with distinct sections (abstract, introduction, methods, results, discussion) where relevant information for specific questions tends to be localized within particular sections. A question about experimental methodology is unlikely to require information from the conclusion, and vice versa.

Semantic chunking effectively exploits this locality by presenting the model with self-contained scientific arguments without the noise of entire papers. The chunking process acts as a precision filter, reducing the cognitive load on the generation model and enabling more focused reasoning. This architectural property

explains why higher-order sections outperformed whole documents across all models on CovidQA. Document length emerges as a critical mediating factor in chunking strategy effectiveness. The average BankReg-IT document (28,109 tokens) is approximately $7.3\times$ longer than the average CovidQA document (3,847 tokens). This length differential has several implications:

1. **Attention distribution**: In longer documents, the "lost in the middle" phenomenon becomes more pronounced, potentially disadvantaging whole-document approaches. However, our results show the opposite for legal texts, suggesting that the benefits of contextual completeness outweigh attention distribution concerns when documents have high internal coherence.

2. **Retrieval precision**: For longer documents, the precision of retrieval becomes more critical. Retrieving an entire 28,000-token legal document when only a specific section is relevant introduces substantial noise. Yet our results indicate that this noise is preferable to the information loss from fragmentation—at least for regulatory texts.

3. **Computational scaling**: The cost differential between chunked and whole-document approaches scales with document length. For CovidQA's shorter documents, whole-document processing is computationally tractable; for BankReg-IT's extensive regulations, the cost difference becomes substantial and may influence architectural decisions in cost-sensitive deployments.

Trade-offs and Recommendations

Based on our comprehensive evaluation, the selection of a chunking strategy must be intrinsically tied to the structural characteristics of the target domain. For high-density texts such as legal, regulatory, and contract documents, preserving long-range semantic dependencies is critical; therefore, whole-document retrieval should be preferred whenever computational resources allow. If context window limitations necessitate segmentation, practitioners should employ the largest feasible chunk sizes and strictly avoid fine-grained, section-level chunking, which our experiments show consistently underperforms. A viable compromise in such scenarios is a hybrid approach, consisting of coarse document-level retrieval followed by a secondary within-document search. Conversely, when processing scientific and technical documentation, information tends to be localized within specific structural boundaries. In these cases, semantic chunking based on higher-order sections (e.g., major document divisions rather than fine-grained paragraphs) provides

the optimal balance, although whole-document approaches remain competitive and may be preferred for their simplicity. For mixed-domain applications, systems should implement adaptive, metadata-driven routing strategies that identify document characteristics before processing, potentially maintaining separate indices with different granularities for different document classes. Model selection introduces meaningful patterns that should further inform deployment decisions. Llama 4 Maverick demonstrated exceptional performance with whole documents in the legal domain but showed greater sensitivity to chunking strategies in the scientific domain, making it highly recommended for applications requiring maximum accuracy on complex, long-form documents where computational cost is secondary. GPT-4.1, on the other hand, exhibited more consistent performance across both domains and chunking strategies, rendering it suitable for applications with heterogeneous document types where per-class strategy optimization is impractical. Finally, Gemini 2.0 Flash provided competitive performance with lower latency, representing an attractive option for high-throughput applications where marginal accuracy differences are acceptable. To operationalize these findings, practical implementations should begin by evaluating domain characteristics—such as document length, internal coherence, and cross-reference density—before selecting a chunking strategy. Furthermore, production systems require continuous monitoring to track performance metrics across document types, enabling strategies to adapt based on observed outcomes. For documents exceeding context limits, hierarchical architectures that retrieve relevant documents before applying within-document search can preserve contextual relationships while maintaining precision. Ultimately, the optimal configuration depends on an explicit balance of cost and accuracy; for example, regulatory compliance applications may justify higher computational costs to ensure accuracy, whereas customer support systems may prioritize throughput and latency. Our findings point toward several promising research directions. Dynamic chunking mechanisms that utilize query analysis to select between whole-document ingestion and granular retrieval based on query complexity represent a particularly compelling opportunity. Self-routing approaches, building on work such as Self-ROUTE [44], could enable systems to adaptively choose strategies in real-time based on both document and query characteristics. Additionally, the development of domain-specific chunking heuristics informed by document structure analysis could improve performance in specialized applications. Legal documents, for instance, might benefit from chunking strategies that preserve definitional relationships and cross-references while still enabling efficient retrieval. The continued expansion of context windows will likely shift the optimal tradeoff points identified in this study. However,

the fundamental insight—that domain characteristics should drive architectural decisions—will remain relevant regardless of model capabilities. RAG systems that adapt their strategies to document and query characteristics will consistently outperform those applying uniform approaches across heterogeneous content.

5.2.3 Source Attribution

In Retrieval-Augmented Generation systems, ensuring that generated text is accurately attributed to its underlying sources is critical for transparency, trustworthiness, and verifiability. Context-attribution—the task of linking LLM-generated sentences with portions of the retrieved input context—represents a fundamental component for responsible AI deployment. This section presents a comprehensive study on post-hoc context-attribution methods, focusing on the comparison between lightweight cross-encoders and Large Language Models in low-annotation industrial settings.

The Context-Attribution Problem

As RAG systems become increasingly prevalent in production environments, the need for reliable context-attribution grows more pressing. When users interact with AI assistants powered by LLMs, they require not only accurate answers but also the ability to verify the information against the original sources. Corroborative context-attribution, which identifies evidence supporting generated statements, is essential for making LLM-generated content more transparent and trustworthy.

The context-attribution task can be formalized as follows: given an answer a generated in a RAG setting and a set of passages $P = \{p_1, p_2, \ldots, p_m\}$ retrieved by the system, the objective is to identify which passages contain the information used to produce the answer. This task presents several challenges:

- **Semantic complexity**: The generated answer often paraphrases or synthesizes information from multiple sources, making direct string matching ineffective.
- **Annotation scarcity**: Obtaining high-quality annotations for context-attribution requires domain expertise and is prohibitively expensive at scale.
- **Granularity trade-offs**: Attribution can be performed at different levels of granularity—from entire passages to individual sentences—each presenting unique challenges.

- **Computational constraints**: Production environments often require efficient inference with limited computational resources.

Two distinct attribution tasks are addressed in this work: *answer-level context-attribution*, which identifies all passages that entail any portion of the answer, and *sentence-level in-line citation*, which requires fine-grained attribution at the sentence level within the generated response.

Semi-Supervised Cross-Encoders for Context-Attribution

While state-of-the-art LLMs exhibit strong performance across various NLP tasks, their deployment for context-attribution presents practical limitations: high computational costs, dependency on proprietary APIs, and the need for complex prompt engineering. Cross-encoders emerge as a practical and efficient alternative, offering advantages in terms of cost, control, and adaptability. Unlike bi-encoders, which compute separate embeddings for queries and passages, cross-encoders jointly encode both inputs, enabling the generation of fine-grained relevance scores. This unified encoding architecture enhances semantic understanding, making cross-encoders particularly suitable for nuanced tasks like context-attribution. Furthermore, when applied to a limited set of candidates, cross-encoders offer fast inference compatible with real-time retrieval pipelines. The `gte-multilingual-reranker-base` model, with approximately 500 million parameters, serves as the baseline cross-encoder in our experiments. To address the inherent complexity of long passages, three inference strategies are explored:

- **Answer Passage strategy (AP)**: This approach feeds the cross-encoder with a concatenation of the full answer and the full passage. The model outputs the probability that the answer is entailed in the passage. After inference, passages are ranked by descending probability and the top-k passages are retained as attributed sources. While this strategy requires only a single model inference per answer-passage pair, it may fail to capture complex information when the context window is saturated.
- **Sentence-Sentence strategy (SS)**: Each sentence from the answer is paired with each sentence from the associated passages, and the model outputs the entailment probability for each pair. This strategy captures more granular semantics but incurs significant computational overhead due to the high number of required inferences. Moreover, chunking passages into sentences might fail to capture long dependencies.

- **Sliding Window strategy (SW)**: A sliding window of w_a sentences for the answer and w_p sentences for the passages is employed. This approach aims to identify the optimal level of granularity while reducing computational overhead compared to the sentence-sentence strategy, though it requires tuning the window hyperparameters. The hyperparameter search is constrained to $2 \leq w_a \leq 4$ and $w_a \leq w_p \leq 8$, based on the hypotheses that passage windows benefit from longer chunks due to long information dependencies and that cross-encoders benefit from extra answer context to disambiguate false positives.

A key contribution of this work is the semi-supervised training approach that leverages synthetic data generation to overcome annotation bottlenecks. Given the scarcity of annotated data in real-world industrial scenarios, synthetic question-answer pairs are generated using a quantized version of LLaMa-3.1-8B-instruct. The generation process follows a structured methodology:

1. **Clustering**: Training passages are embedded using an off-the-shelf bi-encoder and clustered into 10 groups using k-means clustering.

2. **Example Selection**: Examples closest to cluster centroids are retained as gold examples for few-shot learning.

3. **Synthetic Generation**: For each passage in the training corpus, two gold examples are sampled and used in a few-shot setting to generate synthetic question-answer pairs.

4. **Fine-tuning**: The cross-encoder is fine-tuned on the synthetically generated data using the following settings: learning rate of 2×10^{-7}, 1 epoch, batch size of 4, maximum sequence length of 2048 tokens, with early stopping based on F1 score at a threshold of 0.9.

To mimic low-resource environments typical of industrial applications, only 10 examples from the training set are used as gold samples. The synthetic data generation is performed using vLLM with prompts filtered to a maximum of 6,000 tokens to avoid out-of-memory errors. A manual review of 30 items per dataset confirmed that the generated queries and answers were meaningful and contextually relevant.

Evaluation Framework

The evaluation framework encompasses four datasets representing different domains and annotation conditions, as summarized in Table 5.3.

Table 5.3: Dataset statistics for context-attribution evaluation.

	Proprietary	**TREC-RAG**	**ASQA**	**ELI5**
Passages	37,016	6,856	487,643	89,115
Avg. Tokens	333.79	220.79	584.70	114.47
Avg. Passages	4.45	8.56	10.00	10.00
Train (Q-A pairs)	27,000	6,000	33,693	86,904
Eval (Q-A pairs)	25	25	25	25
Hyp (Q-A pairs)	13	25	25	25
Test (Q-A pairs)	12	30	200	200

The **Proprietary Legal Corpus** consists of a diverse collection of legal documents supplied by an industrial partner, comprising approximately 37,000 pages of Italian financial regulations and internal corporate materials (including emails and internal communications). This dataset is fully annotated by domain experts (legal professionals with degrees in legal fields) and enables the computation of standard precision, recall, and F1 metrics.

TREC-RAG is originally designed to evaluate factoid QA systems. The RAG variant repurposes the corpus for retrieval-based LLM generation by pairing questions with relevant evidence passages. The retrieval corpus consists of Wikipedia articles segmented into fixed-length passages. We adopt a subset from the Ragnarök framework containing 100 retrieved passages for each question, with the top-20 reranked passages used for annotation and inference. The dataset contains answers generated with GPT-4. **ASQA** is a dataset for long-form factoid question answering focusing on ambiguous questions requiring nuanced responses. The retrieval corpus is constructed from a 2018 Wikipedia dump, with documents split into 800-word passages with a 200-word stride. **ELI5** is a long-form question answering dataset derived from the Reddit forum "Explain Like I'm Five," paired with passages from the Sphere corpus. For the proprietary dataset and TREC-RAG with full annotation, standard precision, recall, and F1 metrics are computed. For ASQA and ELI5, where budget constraints limited annotation, modified versions of citation precision (p_c), citation recall (r_c), and citation F1 ($f1_c$) are employed, computing information entailment rather than sentence or answer entailment.

Let $I = \{i_1, i_2, \ldots, i_n\}$ represent the atomic information units extracted from the answer using a NLI model, and let $e_{q,j} \in \{0, 1\}$ denote whether information i_q is entailed by passage p_j. Citation recall is defined as:

$$r_c = \frac{|I_{ent}|}{|I|} \tag{5.2}$$

where $I_{ent} = \{i_q \mid \exists j : e_{q,j} = 1\}$ represents the set of information units entailed by at least one attributed passage. Citation precision is defined as:

$$p_c = \frac{|P_{ent}|}{|P|} \tag{5.3}$$

where $P_{ent} = \{p_j \mid \exists q : e_{q,j} = 1\}$ represents the set of passages that entail at least one information unit from the answer. GPT-4o is employed for both information extraction and entailment inference. To establish a comprehensive baseline, the cross-encoder approach is compared against both open-source (LLaMa-3.1-8B-instruct) and proprietary (GPT-4o) language models. Two prompting strategies are evaluated: A straightforward prompt that concatenates instructions with the question-passage-answer triplet, instructing the model to respond with "Yes" or "No" to indicate whether the passage entails the answer. A more sophisticated prompt that guides the model through a threefold reasoning process: (i) identifying common information between the answer and passage, (ii) reasoning about information entailment, and (iii) determining the final entailment verdict. The model outputs a structured JSON with three keys for transparency and interpretability.

Results and Discussion

Table 5.4 presents the answer-level context-attribution results on the proprietary legal dataset. The cross-encoder with the sliding-window strategy achieves an F1 score of 84.15 with the frozen model (GTE) and 87.55 after fine-tuning on synthetic data ($\text{GTE}_{\text{tuned}}$). Among LLMs, GPT-4o with the CoT prompt achieves an F1 of 87.27, driven primarily by exceptional recall (96.00). However, the fine-tuned cross-encoder achieves superior performance while offering significant advantages in terms of deployment cost and latency. The smaller LLaMa model exhibits poor performance with the lowest scores across all metrics. Table 5.5 presents the results on TREC-RAG, a dataset that requires handling long dependencies for correct attribution.

On TREC-RAG, the fine-tuned cross-encoder ($\text{GTE}_{\text{tuned}}$) with SW strategy achieves 77.41 F1, comparable to GPT-4o with CoT (78.74). Notably, the baseline

Table 5.4: Answer-level context-attribution metrics on the proprietary legal dataset. Parameters are reported as top-k/answer_sw/passage_sw/score_threshold.

Model	Strategy	Precision	Recall	F1
GTE	SW	84.76	85.67	84.15
GTE	SS	76.76	92.00	83.64
GTE	AP	75.00	72.00	73.47
GTE_{tuned}	SW	<u>85.88</u>	89.33	**87.55**
GTE_{tuned}	SS	76.76	92.00	83.64
GTE_{tuned}	AP	84.62	88.00	<u>86.27</u>
GPT-4o	CoT	80.00	**96.00**	87.27
GPT-4o	Baseline	**90.48**	76.00	82.61
LLaMa-3.1	Baseline	69.23	72.00	70.59

Table 5.5: Answer-level context-attribution metrics on TREC-RAG. Parameters are reported as top-k/answer_sw/passage_sw/score_threshold.

Model	Strategy	Precision	Recall	F1
GTE	SW	74.08	82.60	75.68
GTE	SS	<u>80.55</u>	48.23	58.78
GTE	AP	70.79	81.73	72.42
GTE_{tuned}	SW	70.32	<u>90.82</u>	<u>77.41</u>
GTE_{tuned}	SS	63.11	**98.61**	76.96
GTE_{tuned}	AP	67.32	80.37	70.05
GPT-4o	CoT	**73.98**	89.29	**78.74**
GPT-4o	Baseline	45.03	29.64	31.17
LLaMa-3.1	Baseline	57.04	42.01	42.05

GPT-4o prompt performs very poorly (31.17 F1), highlighting the importance of prompt engineering for LLM-based attribution. The SS strategy with $\text{GTE}_{\text{tuned}}$ achieves exceptional recall (98.61) but lower precision due to long dependencies and non-contextualized passages. Table 5.6 presents the context-attribution performance on ASQA using citation metrics computed via NLI. On ASQA,

Table 5.6: Answer-level context-attribution metrics on ASQA. Parameters are reported as top-k/answer_sw/passage_sw/score_threshold.

Model	**Strategy**	p_c	r_c	$f1_c$
GTE	SW	<u>91.48</u>	69.16	78.77
GTE	SS	81.38	69.57	75.06
GTE	AP	88.96	68.27	77.25
$\text{GTE}_{\text{tuned}}$	SW	88.76	<u>70.05</u>	78.31
$\text{GTE}_{\text{tuned}}$	SS	85.44	66.66	74.90
$\text{GTE}_{\text{tuned}}$	AP	90.27	69.34	<u>78.47</u>
GPT-4o	CoT	**96.90**	66.73	**79.03**
GPT-4o	Baseline	96.09	59.54	73.56
LLaMa-3.1	Baseline	93.96	52.11	67.14

hyperparameter tuning improves model performance across all configurations. GPT-4o with CoT achieves the highest $f1_c$ (79.03) with exceptional precision (96.90), while cross-encoders achieve competitive scores with significantly lower computational costs. The SW strategy with GTE achieves the highest recall among cross-encoders (70.05 for $\text{GTE}_{\text{tuned}}$). Table 5.7 presents the results on ELI5, where overall performance is notably lower than on other datasets, reflecting the inherent difficulty of this benchmark.

On ELI5, cross-encoders consistently outperform LLMs in terms of $f1_c$, with GTE achieving the highest score (25.76). While LLMs achieve higher precision, their recall is significantly lower, resulting in poor overall performance. This highlights the advantage of cross-encoders on challenging benchmarks where LLMs struggle to identify relevant passages. For sentence-level attribution, the evaluation is conducted by inferring information entailment on 8-sentence passages with a stride of 2 sentences, with top-k fixed to 20 for all cross-encoders. The model tuned on synthetic data exhibits smoother behavior compared to the frozen counterpart. $\text{GTE}_{\text{tuned}}$ consistently outperforms both LLaMa and the untuned GTE across all datasets. GPT-4o with both prompt variants shows similar behavior with

Table 5.7: Answer-level context-attribution metrics on ELI5. Parameters are reported as top-k/answer_sw/passage_sw/score_threshold.

Model	**Strategy**	p_c	r_c	$f1_c$
GTE	SW	51.32	<u>17.20</u>	<u>25.76</u>
GTE	SS	47.28	16.91	24.91
GTE	AP	49.54	14.94	22.96
GTE_{tuned}	SW	<u>57.96</u>	16.09	25.19
GTE_{tuned}	SS	49.88	**16.97**	24.92
GTE_{tuned}	AP	53.61	15.17	23.65
GPT-4o	CoT	81.16	12.46	21.60
GPT-4o	Baseline	**90.08**	5.85	10.99
LLaMa-3.1	Baseline	71.83	9.12	16.19

slightly better scores for the CoT version, and is rarely outperformed by other models. Regardless of the fact that cross-encoders were not trained for sentence-level entailment, tuning on synthetically generated data slightly improves attribution capabilities. For in-line citations, the comparison favors LLMs that can obtain high scores with complex prompting strategies. Manual evaluation of 30 randomly sampled items per dataset reveals several patterns: (i) pretrained retrievers often misclassify similar-looking passages, (ii) tuned retrievers improve but still occasionally misclassify, (iii) LLaMa relies heavily on surface cues and is sometimes misled by key entities, (iv) GPT-4o may overlook marginally mentioned but crucial information, and (v) CoT can still make reasoning errors, especially with implicit subject shifts. The tuned cross-encoder averages 2.15s/query (0.6s/query with parallelization), yielding approximately 1,674 queries/hour. With a Tesla T4 at $0.526/hour (AWS), this results in approximately $0.31 per 1,000 queries. In contrast, LLM inference with GPT-4o on 1,000 queries with approximately 4M tokens costs approximately $10. Parallelism improves LLM throughput but does not close the cost gap, making cross-encoders approximately 32$\times$ more cost-effective.

Key Findings and Industrial Implications

The experimental analysis yields several important findings with direct implications for industrial RAG deployments. While Large Language Models are often

the default choice for many NLP tasks, their effectiveness for context-attribution varies significantly. For instance, open-source LLMs like LLaMa-3.1-8B, despite their strength in other areas, are generally not well-suited for this specific task. Conversely, proprietary models such as GPT-4o provide consistently strong performance for fine-grained sentence-level attribution; however, they become inconsistent on answer-level attribution, losing effectiveness as the generated answer length increases. Advanced prompt engineering can mitigate some of these LLM limitations and improve overall attribution performance, provided the model is capable of structured data generation. The impact of prompting is substantial: on the TREC-RAG dataset, a baseline GPT-4o prompt performs very poorly (achieving a 31.17 F1 score) compared to a Chain-of-Thought approach (78.74 F1). Given these constraints and the high computational overhead of LLMs, small cross-encoders emerge as highly valid alternatives for post-generation answer-level context-attribution. Their performance can be effectively optimized by training on synthetically generated data, allowing for robust deployment even in scenarios where manually annotated data is scarce or expensive to obtain. When paired with a sliding-window strategy, cross-encoders offer the optimal balance between attribution performance and computational efficiency across most datasets. Crucially, these cross-encoders achieve comparable or even superior performance to top-tier LLMs at a fraction of the computational cost. By offering approximately 32x cost savings compared to GPT-4o, they represent a highly scalable solution for high-volume production environments. Ultimately, these findings underscore the practicality and promise of semi-supervised cross-encoders for robust, interpretable, and resource-efficient context-attribution, particularly within production-oriented or specialized domains constrained by limited computational resources and annotated data.

Chapter 6

Conclusions and Future Perspectives

This thesis has presented a comprehensive investigation into improving Retrieval-Augmented Generation (RAG) pipelines for question answering over documents. Through systematic empirical studies and the development of novel methodologies, we have addressed critical challenges across the three fundamental components of modern RAG systems: information retrieval, question answering with large language models (LLMs), and system-level optimization, including chunking strategies and source attribution. This concluding chapter synthesizes the contributions made throughout this work, discusses their scientific and industrial impact, acknowledges prevailing limitations, and outlines promising directions for future research.

6.1 Summary of Contributions and Impact

The contributions of this thesis span both theoretical advancements and practical industrial applications, reflecting the dual nature of this industry-collaborative doctoral program. By moving away from fragmented optimizations, this work proposes a holistic approach to RAG system enhancement.

6.1.1 Scientific Advancements

A primary focus of this research has been the systematic evaluation and enhancement of Information Retrieval (IR) models. As detailed in Chapter 3, our comprehensive benchmarking of embedding models across English and Italian addressed a significant gap in multilingual IR evaluation. We demonstrated that state-of-the-art multilingual models achieve competitive performance across languages, consistently outperforming language-specific alternatives. Notably, our findings suggest that architectural choices often outweigh mere parameter count in determining model efficacy. To tackle the persistent challenge of data scarcity in specialized domains, we subsequently introduced the SAGE (Synthetic Augmentation for Guided Embeddings) framework. By leveraging LLMs to generate high-quality training data from unlabeled corpora, SAGE enables domain-adaptive fine-tuning that substantially outperforms general-purpose baselines, offering a modular architecture conducive to continuous learning.

In the realm of Question Answering, Chapter 4 established a multi-dimensional evaluation framework that goes beyond traditional metrics. Our systematic analysis across diverse datasets highlighted a critical divergence between semantic and syntactic metrics, confirming the inadequacy of purely lexical evaluation for generative QA. More importantly, we identified a consistent "groundedness gap," revealing that current models frequently prioritize fluency over faithfulness to source material. This investigation naturally extended to tabular data, where we demonstrated that HTML representations consistently outperform plain text formats across all tested models. Furthermore, our experiments established that Pandas-based semantic parsing significantly outperforms SQL-based approaches for complex financial tables.

Finally, we addressed critical system-level optimizations that bridge the gap between retrieval and generation. Chapter 5 provided evidence-based guidance on document segmentation in the era of long-context LLMs, revealing that optimal chunking strategies are highly domain-dependent. While structured scientific content benefits from fine-grained semantic chunking, complex legal documents with intricate cross-references favor whole-document retrieval. Complementing this, we developed a semi-supervised cross-encoder approach for source attribution. Trained on synthetically generated data, this methodology achieves performance comparable to state-of-the-art proprietary LLMs but at a fraction of the computational cost, enabling efficient, verifiable QA deployment.

6.1.2 Industrial Value and State-of-the-Art Advancement

Beyond scientific inquiry, this thesis has generated tangible value in production environments. We introduced the RAG-Dataset format, an extension of the BEIR schema, which standardizes multi-level relevance annotations and provenance tracking. This format serves as the backbone for an automated continuous improvement pipeline that we successfully deployed across financial, legal, and compliance domains. By integrating retriever fine-tuning, synthetic data generation, and active learning, this pipeline transforms static deployments into continuously learning systems, ensuring sustained quality gains without proportional increases in human annotation costs.

Furthermore, the deployment of our semi-supervised source attribution methodology has enabled compliance with strict regulatory requirements for traceability, opening new operational domains that were previously constrained by verification bottlenecks. Collectively, these contributions advance the state of the art by providing scalable, evidence-based solutions to bridge the gap between general-purpose models and the stringent demands of specialized enterprise applications.

6.2 Limitations and Navigating Trade-offs

Despite these advancements, several open challenges and inherent system trade-offs require careful navigation by system designers.

Foremost among these challenges is the persistent risk of hallucination. While RAG systems significantly mitigate ungrounded generation compared to pure LLM approaches, the documented "groundedness gap" indicates that absolute factual accuracy cannot yet be guaranteed. This limitation is compounded when systems operate across linguistic or domain boundaries. We observed substantial performance degradation in cross-lingual scenarios and when shifting from general to highly specialized domains (such as medical or complex legal texts). While targeted adaptation frameworks like SAGE offer a viable mitigation strategy, they still require computational resources and domain-specific corpora that may not always be accessible. Furthermore, the reliance on reference-based metrics highlights an ongoing struggle: developing reliable, automatic evaluation methods without gold-standard answers remains a significant hurdle for scaling QA systems in niche domains.

Designing effective RAG systems also involves managing fundamental trade-offs. For instance, optimizing retrieval granularity often conflicts with preserving

context coherence; finer chunking improves retrieval precision but risks fragmenting the semantic dependencies crucial for understanding complex documents. Similarly, there is a constant tension between computational cost and generation quality. While long-context processing and LLM-based attribution yield high-fidelity results, their operational costs are prohibitive for many applications. Strategic chunking and smaller, specialized architectures offer cost-effective alternatives, but they demand rigorous, domain-specific tuning. Finally, while synthetic data generation enables scalable adaptation, balancing it with human annotation is critical to prevent the introduction of systematic biases that expert reviewers would otherwise catch.

6.3 Future Directions

The convergence of expanded context windows, sophisticated retrieval methods, and more capable generation models paves the way for several promising research trajectories.

A critical technological frontier is the transition toward unified multimodal RAG pipelines. Real-world documents are rarely plain text; integrating vision-language models to seamlessly process text, tables, figures, and complex layouts within a single framework will address one of the most significant limitations of current systems. Alongside multimodal integration, future systems should focus on adaptive retrieval strategies that dynamically adjust their methods—such as switching between dense, sparse, or hybrid retrieval, or altering chunking granularity—based on real-time query analysis. As context windows expand, investigating hybrid architectures that intelligently combine targeted retrieval with long-context processing will be essential for balancing efficiency and comprehensive document understanding.

From a research perspective, closing the groundedness and cross-lingual performance gaps remains paramount. Future work should explore constrained decoding methods that actively verify claims against retrieved context during generation, as well as attribution-aware training objectives. Additionally, extending RAG capabilities to support interactive, multi-turn conversations with robust context management and dialogue state tracking will make these systems more natural and effective. Finally, developing comprehensive explainability mechanisms will be crucial for enhancing user trust, ensuring that users understand not just the answer, but the provenance and reliability of the underlying sources.

6.4 Concluding Remarks

This thesis demonstrates that substantial improvements in RAG pipeline performance are achievable through the systematic, holistic optimization of its constituent components. The contributions presented herein range from foundational empirical insights—such as the efficacy of multilingual models and the domain-dependency of chunking strategies—to robust, deployable methodologies that bridge the gap between academic research and industrial application.

Progress in natural language processing requires a dual commitment to rigorous scientific methodology and practical operational constraints. The benchmarking frameworks established in this work provide reproducible baselines for future research, while our continuous improvement pipelines offer immediately actionable solutions for practitioners. As large language models continue to evolve, the necessity of grounding their generative capabilities in reliable, verifiable sources only grows more acute. The foundations laid by this research provide essential building blocks for the next generation of intelligent document processing systems, ensuring they can reliably extract, synthesize, and communicate knowledge from our ever-expanding digital corpus.

Bibliography

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Rami Aly, Zhijiang Tang, Samson Tan, and George Karypis. Learning to generate answers with citations via factual consistency models. *arXiv preprint arXiv:2406.13124*, 2024.

[3] Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems*, 20(4):357–389, 2002.

[4] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2024.

[5] Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Massimiliano Ciaramita, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, et al. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*, 2022.

[6] Giovanni Bonisoli, Maria Pia Di Buono, Laura Po, and Federica Rollo. Dice: a dataset of italian crime event news. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2985–2995, New York, NY, USA, 2023. Association for Computing Machinery.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[8] Sebastian Bruch, Siyu Gai, and Amir Ingber. An analysis of fusion functions for hybrid retrieval. *ACM Transactions on Information Systems*, 42(1):1–35, 2023.

[9] Jannis Bulian, Christian Buck, Wojciech Gajewski, Benjamin Boerschinger, and Tal Schuster. Tomayto, tomahto. beyond token-level answer equivalence for question answering evaluation. *arXiv preprint arXiv:2202.07654*, 2022.

[10] Ben Carterette and Ellen M Voorhees. Overview of information retrieval evaluation. In *Current Challenges in Patent Information Retrieval*, pages 69–85. Springer, 2011.

[11] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 2024.

[12] Kunal Chen et al. Blended RAG: Improving RAG accuracy with semantic search and hybrid query-based retrievers. *arXiv preprint arXiv:2404.07220*, 2024.

[13] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[14] Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 758–759, 2009.

[15] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *arXiv preprint arXiv:2305.14314*, 2023.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

[17] Li Dong and Mirella Lapata. Language to logical form with neural attention. *ArXiv*, abs/1601.01280, 2016.

[18] Kenneth Enevoldsen et al. MMTEB: Massive multilingual text embedding benchmark. *arXiv preprint arXiv:2502.13595*, 2025.

[19] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, 2024.

[20] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. SPLADE v2: Sparse lexical and expansion model for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292, 2022.

[21] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. SPLADE: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292, 2021.

[22] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. RARR: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*, 2023.

[23] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. *arXiv preprint arXiv:2212.10496*, 2023.

[24] Luyu Gao, Yunyi Yao, Yichao Chen, Chao Xu, Yuxian Fan, Haomin Zhao, and Jamie Callan. Scaling deep contrastive learning batch size under

memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP*, pages 316–321, 2021.

[25] Gemma Team. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[26] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pages 3929–3938. PMLR, 2020.

[27] Kailash A Hambarde and Hugo Proença. Information retrieval: Recent advances and beyond. *IEEE Access*, 11:76581–76604, 2023.

[28] Matthew Henderson, Pawel Budzianowski, Iñigo Casanueva, Sam Coope, Daniela Gerz, Girish Kumar, Nikola Nikolic, Georgios Spithourakis, Pei-Hao Su, Ivan Vulić, et al. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*, 2017.

[29] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[30] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[31] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Real-time inference in multi-sentence tasks with deep pretrained transformers. *arXiv preprint arXiv:1905.01969*, 2019.

[32] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[33] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *ArXiv*, abs/1705.03551, 2017.

[34] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6769–6781. Association for Computational Linguistics, 2020.

[35] Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48, 2020.

[36] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[37] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213, 2022.

[38] Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. Neural semantic parsing with type constraints for semi-structured tables. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[39] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

[40] Dawn Lawrie, Eugene Yang, Douglas W Oard, and James Mayfield. Neural approaches to multilingual information retrieval. *arXiv preprint arXiv:2209.01335*, 2023.

[41] Jinhyuk Lee, Zhuyun Dai, Sai Meher Karthik Duddu, Tao Lei, Iftekhar Naim, Ming-Wei Chang, and Vincent Y Zhao. Rethinking the role of token retrieval in multi-vector retrieval. *Advances in Neural Information Processing Systems*, 36, 2024.

[42] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020.

[43] Kuan Li et al. Lara: Benchmarking retrieval-augmented generation and long-context llms – no silver bullet for lc or rag routing. *arXiv preprint arXiv:2502.09977*, 2025.

[44] Xinze Li et al. Long context vs. rag for llms: An evaluation and revisits. *arXiv preprint arXiv:2501.01880*, 2024.

[45] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *ArXiv*, abs/2308.03281, 2023.

[46] Zhuowan Li et al. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. *arXiv preprint arXiv:2407.16833*, 2024.

[47] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics, 2004.

[48] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[50] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[51] Xueguang Ma, Liang Zhang, and Jimmy Lin. Fine-tuning LLaMA for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425, 2024.

[52] Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.

[53] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, 2023.

[54] Matteo Muffo and Elisa Bertino. BERTino: An italian DistilBERT model. *arXiv preprint arXiv:2303.18121*, 2023.

[55] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[56] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019.

[57] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*, 2020.

[58] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*, 2019.

[59] Ermelinda Oro, Francesco Maria Granata, Antonio Lanza, Amir Bachir, Luca De Grandis, and Massimo Ruffolo. Evaluating retrieval-augmented generation for question answering with large language models. In *Ital-IA 2024: 4th National Conference on Artificial Intelligence, organized by CINI*, Naples, Italy, 2024.

[60] Ermelinda Oro, Francesco Maria Granata, and Massimo Ruffolo. A comprehensive evaluation of embedding models and LLMs for IR and QA across english and italian. *Big Data and Cognitive Computing*, 9(141), 2025.

[61] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex

Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, 2022.

[62] 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*, pages 311–318, 2002.

[63] Jay M Ponte and W Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[64] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. RankZephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv preprint arXiv:2312.02724*, 2023.

[65] Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*, 2024.

[66] Renyi Qu, Ruixuan Tu, and Forrest Bao. Is semantic chunking worth the computational cost? *arXiv preprint arXiv:2410.13070*, 2024.

[67] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Preprint*, 2018.

[68] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[69] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[70] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. *arXiv preprint arXiv:1806.03822*, 2018.

[71] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[72] Revanth Gangi Reddy, JaeYoung Doo, Yifei Xu, Md Arafat Sultan, Avirup Sil, Heng Ji, and Shih-Fu Chang. FIRST: Faster improved listwise reranking with single token decoding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

[73] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *ArXiv*, abs/1908.10084, 2019.

[74] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

[75] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988.

[76] Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. PLAID: An efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pages 1747–1756, 2022.

[77] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, 2022.

[78] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

[79] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? Investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, 2023.

[80] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27, 2014.

[81] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*, 2021.

[82] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[83] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[84] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[85] Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360, 2022.

[86] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2024.

[87] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, 2023.

[88] Xiaohua Wang et al. Searching for best practices in retrieval-augmented generation. *arXiv preprint arXiv:2407.01219*, 2024.

[89] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

[90] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

[91] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022.

[92] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian yun Nie. C-pack: Packed resources for general chinese embeddings. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.

[93] Xinping Xiao, Yujia Wang, Hao Zhang, Hang Li, and Min Zhang. FunnelRAG: A coarse-to-fine progressive retrieval paradigm for RAG. *arXiv preprint arXiv:2410.10293*, 2024.

[94] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*, 2020.

[95] Peng Xu et al. Retrieval meets long context large language models. *arXiv preprint arXiv:2310.03025*, 2024.

[96] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*, 2024.

[97] Soyoung Yoon, Eunbi Hwang, Jinhyuk Cho, Jaewoo Lee, and Minjoon Lee. ListT5: Listwise reranking with fusion-in-decoder improves zero-shot retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 9292–9305, 2024.

[98] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 1–9. Association for Computational Linguistics, 2022.

[99] Jiajie Zhang, Yushi Bai, Xin Lv, Wanjun Gu, Danqing Liu, Minhao Zou, Shulin Cao, Lei Hou, Yuxiao Dong, Ling Feng, et al. LongCite: Enabling LLMs to generate fine-grained citations in long-context QA. *arXiv preprint arXiv:2409.02897*, 2024.

[100] Tianjun Zhang, Shishir G Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E Gonzalez. RAFT: Adapting language model to domain specific RAG. *arXiv preprint arXiv:2403.10131*, 2024.

[101] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. *arXiv preprint arXiv:1904.09675*, 2020.

[102] Jihao Zhao et al. Moc: Mixtures of text chunking learners for retrieval-augmented generation system. *arXiv preprint arXiv:2503.09600*, 2025.

[103] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021.

[104] Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. RankT5: Fine-tuning T5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313, 2023.