A Systematic Review of Key Retrieval-Augmented Generation (RAG) Systems: Progress, Gaps, and Future Directions

Agada Joseph Oche*[1], Ademola Glory Folashade†[1], Tirthankar Ghosal[2], and Arpan Biswas[3]

[1]Bredesen Center for Interdisciplinary Research, University of Tennessee, Knoxville, USA, 37996
[2]National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, USA, 37830
[3]University of Tennessee-Oak Ridge Innovation Institute, University of Tennessee, Knoxville, USA, 37996

July 28, 2025

arXiv:2507.18910v1 [cs.CL] 25 Jul 2025

Abstract

Retrieval-Augmented Generation (RAG) represents a major advancement in natural language processing (NLP), combining large language models (LLMs) with information retrieval systems to enhance factual grounding, accuracy, and contextual relevance. This paper presents a comprehensive systematic review of RAG, tracing its evolution from early developments in open-domain question answering to recent state-of-the-art implementations across diverse applications. The review begins by outlining the motivations behind RAG, particularly its ability to mitigate hallucinations and outdated knowledge in parametric models. Core technical components—retrieval mechanisms, sequence-to-sequence generation models, and fusion strategies—are examined in detail. A year-by-year analysis highlights key milestones and research trends, providing insight into RAG's rapid growth. The paper further explores the deployment of RAG in enterprise systems, addressing practical challenges related to retrieval of proprietary data, security, and scalability. A comparative evaluation of RAG implementations is conducted, benchmarking performance on retrieval accuracy, generation fluency, latency, and computational efficiency. Persistent challenges such as retrieval quality, privacy concerns, and integration overhead are critically assessed. Finally, the review highlights emerging solutions, including hybrid retrieval approaches, privacy-preserving techniques, optimized fusion strategies, and agentic RAG architectures. These innovations point toward a future of more reliable, efficient, and context-aware knowledge-intensive NLP systems.

Keywords: Retrieval Augmented Generation (RAG), Large Language Model (LLM), Generative AI, Natural Language Model (NLP)

*Corresponding author: joe88data1@gmail.com
†gloryademola112@gmail.com

1 Introduction

Since its formal introduction in the seminal work of [52] in 2020, Retrieval-Augmented Generation (RAG) has witnessed rapid advancements, marked by a significant surge in research interest and scholarly publications. This paper offers a unique and comprehensive review of the key developments and contributions in the field to date. The remainder of this introduction outlines the background and motivation for this review, defines its scope and objectives, and provides an overview of the paper's organization.

1.1 Background and Motivation

Large-scale pre-trained language models have demonstrated an ability to store vast amounts of factual knowledge in their parameters, but they struggle with accessing up-to-date information and providing verifiable sources. This limitation has motivated techniques that augment generative models with information retrieval. *Retrieval-Augmented Generation* (RAG) emerged as a solution to this problem, combining a neural retriever with a sequence-to-sequence generator to ground outputs in external documents [52]. The seminal work of [52] introduced RAG for knowledge-intensive tasks, showing that a generative model (built on a BART encoder–decoder) could retrieve relevant Wikipedia passages and incorporate them into its responses, thereby achieving state-of-the-art performance on open-domain question answering. RAG is built upon prior efforts in which retrieval was used to enhance question answering and language modeling [48, 26, 45]. Unlike earlier extractive approaches, RAG produces free-form answers while still leveraging non-parametric memory, offering the best of both worlds: improved factual accuracy and the ability to cite sources. This capability is especially important to mitigate *hallucinations* (i.e., believable but incorrect outputs) and to allow knowledge updates without retraining the model [52, 33].

Since its introduction, RAG has gained significant attention in both research and industry. A growing body of literature has extended RAG with improved retrievers and generators, and the approach has been applied to a wide range of domains. By 2023, the RAG paradigm underpinned hundreds of research publications and numerous commercial systems [21, 60]. In academia, researchers have scaled up retrieval-augmented models and refined their architectures—examples include leveraging larger pre-trained models with retrieval in the loop [e.g., 35, 10]. In parallel, industry adoption of RAG has been swift: leading tech companies have integrated retrieval-augmented generators into search engines, virtual assistants, and enterprise question-answering applications [33, 60]. RAG now powers applications from open-domain QA and customer support chatbots to tools that automatically generate answers with supporting evidence. This broad adoption underscores the significance of RAG as a foundation for making generative AI more reliable and knowledge-aware. This paper provides a unique perspective on to review of literature in RAG by providing detailed yearly research progress in RAG, developing new perspectives, and evaluating trends.

1.2 Scope and Objectives

The objective of this systematic review is to provide a comprehensive overview of the development of RAG and its expanding role in information access. We aim to answer several key research questions: (1) *How has RAG been progressed every year since its inception, and what are the major technical milestones in its research and deployment?* (2) *What challenges and solutions have emerged for integrating RAG with proprietary or private data sources, and what gaps remain (e.g., in security and privacy)?* (3) *How has RAG been used in accelerating material discovery and characterization* (4) *How are RAG systems categorized and how does this categorization affect their performance?*. By addressing these questions, the review seeks to chart the evolution of RAG, evaluate its current capabilities and limitations, and identify areas for future work.

Over the past few years, progress in RAG has been marked by continuous innovation and new applications. We chronicle the advancement year-by-year, highlighting important academic contributions and industry developments that have shaped the field. Special attention is given to the integration of RAG with *proprietary data*—an area of growing interest as organizations apply RAG to internal knowledge bases. This involves examining techniques for efficient retrieval on private corpora and the handling of sensitive information, as well as open issues around data privacy [101]. Recent systems have also demonstrated that users can interact with an RAG-powered agent to obtain information directly from the web or a document corpus, rather than through traditional ranked search results [62, 81]. This paradigm blurs the line between search engine and dialogue agent, opening questions about usability,

accuracy, and trust in such interfaces. Overall, the review considers this and also consolidates knowledge on how RAG techniques have matured and what objectives remain for future research and development.

1.3 Paper Organization

The remainder of this paper is structured as follows. **Section 2 (Methodology)** explains the review methodology, including the literature search strategy, inclusion/exclusion criteria, and approach to data synthesis. **Section 3 (Foundations of RAG)** provides a technical overview of retrieval-augmented generation, describing its core components (retrievers, indexes, generators) and the baseline architectures introduced by seminal works. **Section 4 (Year-by-Year Progress)** presents a chronological synthesis of RAG developments from 2017 onward, highlighting key research milestones. **Section 5 (RAG for Proprietary data and Industry Implementation)** examines enterprise implementation of RAG on proprietary data by key industry players. **Section 6 (RAG Systems Evaluation)** benchmarks different RAG implementations and variants, summarizing their performance across standard datasets and tasks. **Section 7 (Challenges of RAG Systems)**, **Section 8 (Discussion and Future Direction)** and **Section 9 (Conclusion)** finally provide current research gaps and potential future directions to expand the applications to various domain problems.

2 Methodology

This section details the systematic review methodology employed to survey RAG papers. It comprises three main steps: (1) designing a **Search Strategy** to capture a wide range of relevant works, (2) defining **Inclusion and Exclusion Criteria** to refine the initial corpus, and (3) implementing a **Data Extraction and Synthesis** process to analyze and consolidate findings.

2.1 Search Strategy

To ensure comprehensive coverage, we searched both academic and industry-focused literature on RAG. Multiple digital libraries were queried, including ACL Anthology, IEEE Xplore, ACM Digital Library, and Google Scholar. We included documents published from 2017 up to the end of mid 2025, covering early "retrieve-and-generate" approaches and more recent RAG-specific techniques.

Keywords and Databases. We used a set of pre-defined keywords, such as "*retrieval-augmented generation (RAG)*," "*dense retrieval*," "*hybrid retrieval LLM*," "*RAG proprietary data*," and "*LLM web search*." These queries captured works ranging from open-domain QA to secure enterprise implementations. Each keyword search was executed on the above-listed databases, resulting in a pool of references that included journal articles, conference papers, technical reports, and white papers.

Initial Screening. A comprehensive list of potentially relevant works was formed by merging all search results and removing duplicates. Abstracts and titles were checked to confirm alignment with the RAG focus. If a work concentrated solely on retrieval or generation in isolation, without discussing how these components integrate, it was set aside for possible exclusion.

2.2 Inclusion and Exclusion Criteria

We next applied a formal screening process to determine which references genuinely contributed insights into RAG. The criteria below guided our decisions:

2.2.1 Inclusion Criteria

- **Relevant to RAG or closely related baselines**: Works that clearly integrated a retriever with a generative language model or used retrieval to supply context to a generator [52].
- **Knowledge-Intensive Tasks**: Studies centered on open-domain QA, fact-checking, knowledge-grounded dialogue, or other tasks benefiting from external document retrieval.
- **Peer-Reviewed or Reputable Sources**: Publications presented at major AI/NLP venues (ACL, NeurIPS, ICML, EMNLP) or recognized industrial R&D labs (e.g., IBM, Meta, NVIDIA) [33, 60].
- **Preprint**: Preprints are also included to broaden the scope of the survey.
- **English Language**: For consistency and to support thorough evaluation, only English texts were included. end

2.2.2 Exclusion Criteria

- **Solely Retrieval or Solely Generation**: Articles focusing strictly on IR techniques or purely generative models without explicit retrieval-augmented integration were not included.
- **Minimal Discussion of RAG**: Any mention of retrieval+generation was peripheral or superficial, lacking substantial results or analyses.
- **Non-Substantive Publications**: Very short abstracts, publicity notes, or materials without verifiable methodology were excluded.
- **Non-English Papers**: Not considered due to feasibility constraints.

Based on these criteria, the initial corpus was refined into a finalized set of documents deemed pertinent to the state-of-the-art in RAG.

2.3 Data Extraction

For each included publication, we collected key information, such as basic bibliographic details, the retrieval method (e.g., dense vs. sparse), the generator architecture (e.g., T5, BART, GPT), and the evaluated tasks or datasets. This allowed us to systematically compare different RAG implementations and their reported performance. We also looked for and extracted information on the challenges facing RAG implementation. The survey is not limited to peer-reviewed journal articles and conference proceedings, preprints, technical reports, and industry white papers were also reviewed. The review covers the application of RAG systems in all domains

Synthesis Process. All extracted details/data were gathered in a central repository, allowing cross-study comparisons. We grouped research outputs by year of publication to track the chronological evolution of RAG, highlighting seminal breakthroughs and subsequent expansions. In line with systematic review principles, we combined both qualitative (themes, research directions) and quantitative (performance figures, latency measures) observations.

Ensuring Reliability. Disagreements during the review were resolved through discussion or by consulting a third reviewer. This final step ensured consistent application of the inclusion/exclusion criteria and reliable data extraction. The data collected then served as the foundation for our analysis in subsequent sections, including discussions on year-by-year progress, enterprise applications,and proposed solutions.

3 Foundations of RAG

3.1 Definition and Key Concepts

Retrieval-Augmented Generation (RAG) : RAG is a framework that combines a neural text **retrieval** module with a text **generation** module to improve the quality of generated responses in knowledge-intensive tasks. Formally, a RAG model augments a sequence-to-sequence (seq2seq) generator with access to an external text corpus (non-parametric memory) via a retriever [52, 45]. Given an input query x, the retriever R selects a small subset of relevant documents $Z = \{z_1, z_2, \ldots, z_K\}$ from a large corpus $\mathcal{C}$ (with $K \ll |\mathcal{C}|$) [45]. The generator then conditions on both the query x and the retrieved documents Z to produce an output y (such as an answer or a descriptive text). Formally, the RAG model can be viewed as a latent variable generative model that defines a probability distribution over outputs y by marginalizing over the retrieved documents z_i:

$$P(y \mid x) \;=\; \sum_{i=1}^{K} P_{\text{ret}}(z_i \mid x)\; P_{\text{gen}}(y \mid x, z_i)\,, \quad (1)$$

where $P_{\text{ret}}(z_i \mid x)$ is the probability of retrieving document z_i given query x (the retriever's output distribution), and $P_{\text{gen}}(y \mid x, z_i)$ is the generator's conditional probability of producing y given x and a particular retrieved document z_i. In practice, $P_{\text{ret}}(z_i \mid x)$ is typically non-zero only for the top-K retrieved items, providing a tractable approximation to the full sum over the corpus [52]. The retriever R itself can be defined as a function $R(x, \mathcal{C}) \to Z$ that takes a query and returns a small subset Z of corpus $\mathcal{C}$ (with $|Z| = K \ll |\mathcal{C}|$) likely to contain information relevant to x [45]. By design, RAG models maintain two kinds of **memory**: a *parametric memory* (the knowledge encoded in the generator's weights) and a *non-parametric memory* (the external text corpus accessed via retrieval) [52]. A standard RAG architecture is illustrated in Figure 1 below. A key distinction between RAG and pure large language model (LLM) generation is the use of this external non-parametric knowledge source at inference time. Traditional LLM-based generation relies solely on the model's internal parameters for knowledge, which can

lead to **hallucinations** and factual inaccuracies when the model's training data does not adequately cover the query's topic [52]. In contrast, RAG explicitly grounds the generation of retrieved documents that serve as up-to-date evidence, enabling the model to generate content supported by those documents. This retrieval step means that RAG's outputs can be more accurate and **factually correct** compared to generation from a standalone LLM, especially for knowledge-intensive queries. Empirically, [52] demonstrates that a RAG model generates more specific and factual responses than a parametric-only generator, since the retrieved text provides verified information that the generator can incorporate. Another benefit is that the knowledge in a RAG system can be easily **updated** by modifying the document index (or corpus) without retraining the generator, addressing the stiffness of LLMs that have fixed knowledge up to their training cutoff date. In summary, RAG introduces a modular architecture where a retrieval component supplies relevant context "just in time" for the generator, marrying the strengths of Information Retrieval (IR) with those of large-scale generation.

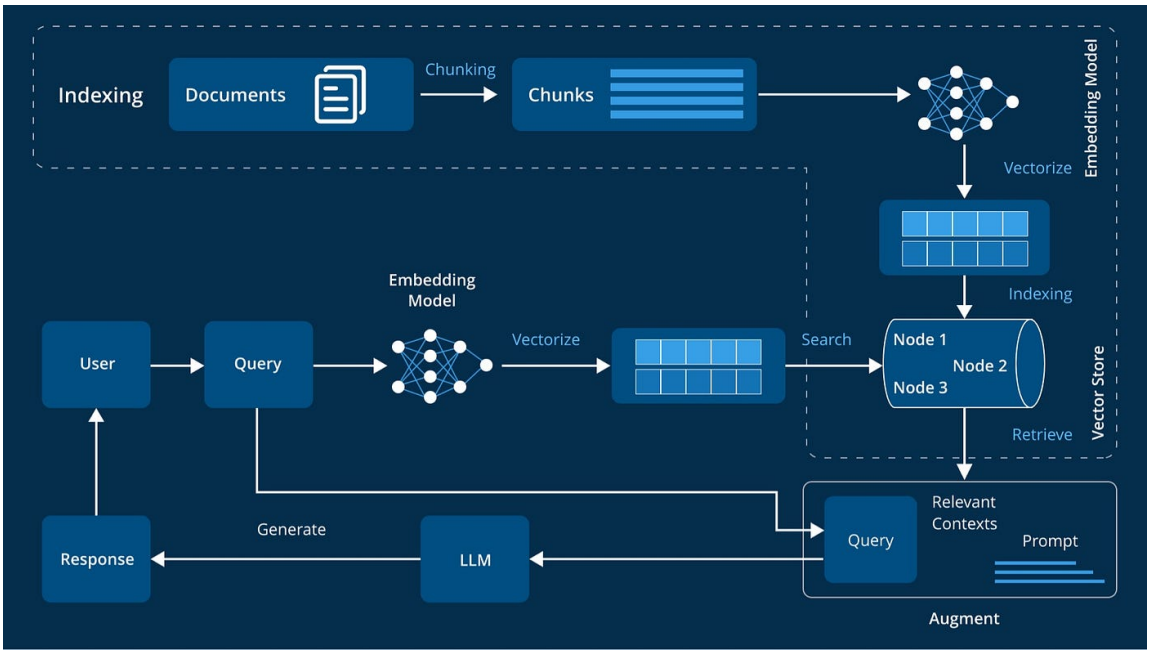

Figure 1: Illustration of a RAG Architecture.

Chunking, Embedding, and (Re)ranking :A typical RAG pipeline consists of four stages: chunking, embedding, (re)ranking, and generation. First, chunking is applied to the knowledge source: large documents are segmented into smaller, self-contained pieces (e.g., paragraphs or passages) for indexing. Using fine-grained text chunks as retrieval units improves the chance that a query will surface a highly relevant fragment, rather than an entire lengthy document [52]. For example, open-domain QA systems split Wikipedia articles into passage chunks to enable pinpoint retrieval of answer-containing segments [45, 52]. The chunk size is typically tuned to balance context completeness and specificity – chunks must be large enough to contain useful context, yet small enough to match queries narrowly and fit within model context windows. Next, each chunk is embedded into a high-dimensional vector representation that encodes its semantic content. This is usually done with a transformer-based bi-encoder that produces dense vector embeddings of text [45]. The embeddings serve as keys in a vector index (or vector database) that supports efficient nearest-neighbor search. At query time, the user's query is likewise embedded into the same vector space, and the system performs similarity search to retrieve the most relevant chunk vectors. In contrast to sparse keyword search, dense embeddings enable semantic matching: a query about "financial earnings" can retrieve a chunk about "quarterly revenue" even if exact words differ [45]. Modern RAG implementations often combine dense retrieval with lightweight filtering or hybrid search (e.g., BM25 + embeddings) to improve recall for difficult queries. The result of the initial retrieval stage is a candidate set of top-k chunks that are potentially relevant to

the query. To further improve precision, an optional re-ranking step is applied on the retrieved candidates before generation. The top-k chunks from the first stage may contain some irrelevant or only tangentially related items, since embedding similarity is a coarse proxy for relevance. A re-ranker model (typically a cross-encoder transformer that jointly encodes query and document) evaluates each retrieved chunk in the context of the query and produces a refined relevance score [63]. By re-scoring and sorting the candidates, the re-ranker ensures that the most pertinent chunks (for example, those actually containing the answer to a question) are ranked highest. This two-stage retrieval process – a fast dense retriever followed by a more accurate but expensive re-ranker – has been shown to significantly boost retrieval performance on knowledge-intensive benchmarks. For instance, neural cross-attention re-rankers achieve substantially higher accuracy than single-stage retrievers alone [63]. In practice, re-ranking is crucial in high-stakes applications (e.g., legal or medical QA), where one must maximize the likelihood that the top context passages truly address the user's query. After re-ranking, the top N (e.g. 3–5) chunks are selected as the final context passages for the generative model. In the generation stage, the LLM produces an answer or response conditioned on the retrieved external chunks. Typically, a sequence-to-sequence model (such as T5 or BART) is used so that the retrieved text can be prepended or incorporated into the model's input along with the user query [52, 73]. During training, the model learns to copy or attend to the relevant facts from the retrieved documents and integrate them into a coherent output. This approach allows the generator to cite up-to-date, specific information beyond its parametric knowledge. For example, [52] show that a RAG model (BART-based) can accurately answer open-domain questions by retrieving and conditioning on Wikipedia text, dramatically reducing hallucinations compared to a standalone LLM. The generated output can also include references to source documents, providing traceability for the facts used. Retrieval augmentation thus serves as a "live memory" for the LLM: it supplies factual grounding from an external knowledge base while the language model creates fluent and contextually relevant text. Notably, recent large-scale studies have demonstrated that even very large models benefit from retrieval augmentation. For instance, the RETRO model augments a 7.5-billion-parameter transformer with a database of trillions of tokens, yielding improved perplexity and factual accuracy by looking up passages during generation [10]. In summary, chunking, embedding, re-ranking, and generation work in concert in RAG systems to leverage external knowledge – the retrieval components identify and prioritize relevant information, and the generation component uses that information to produce answers that are both informative and grounded in source data. This modular design has become a foundation for building more reliable and explainable AI assistants in knowledge-intensive domains.

3.2 Technical Components of RAG

A RAG system is composed of two primary components – a **retriever module** and a **generator module** – along with a strategy for fusing their outputs. We break down these components and the underlying mechanics as follows.

Retriever Module (Dense Passage Retrieval). The retriever's job is to efficiently identify which pieces of text in a large corpus are relevant to the input query x. Modern RAG implementations typically use **dense retrievers** like Dense Passage Retrieval (DPR) [45] in lieu of traditional keyword search. In DPR, a bi-encoder architecture is employed: a *question encoder* $E_q(x)$ maps the query x to a d-dimensional vector, and a *passage encoder* $E_p(d)$ maps each candidate document (or passage) d in the corpus to a d-dimensional vector in the same space. Relevance is assessed via a similarity score, usually the dot product of these vectors:

$$\mathrm{sim}(x, d) \;=\; E_q(x)^\top \, E_p(d)\,.$$

At query time, the retriever computes $v_q = E_q(x)$ and then finds the top-K documents whose vector $E_p(d)$ has highest inner product with v_q. This search for maximum inner product can be implemented efficiently using Approximate Nearest Neighbor techniques (e.g., FAISS) to handle millions of documents in sub-linear time. The output of the retriever is the set $Z = \{z_1, \ldots, z_K\}$ of top-ranked documents and potentially their similarity scores. One can view the retriever as defining a probability distribution $P_{\mathrm{ret}}(z \mid x)$ over documents z in the corpus, such that:

$$P_{\mathrm{ret}}(z \mid x) \;\propto\; \exp\left(E_q(x)^\top E_p(z)\right), \qquad (2)$$

with the normalization $\sum_{d \in \mathcal{C}} \exp(E_q(x)^\top E_p(d))$ (in practice approximated by summing over retrieved candidates rather than all of $\mathcal{C}$). In other words, the retriever assigns higher probability (or score)

to documents whose embedding is most similar to the query's embedding. The DPR retriever is usually first trained on pairs of questions and relevant passages to ensure E_q and E_p produce representations that maximize dot-products for true Q–A pairs and minimize them for irrelevant pairs. The training objective is often a **contrastive loss**: for a given question q with a gold relevant passage p^+ and a set of negative passages $\{p_1^-, \ldots, p_N^-\}$, the encoder is trained to maximize $\text{sim}(q, p^+)$ while minimizing $\text{sim}(q, p^-)$ for negatives. This can be formulated as a cross-entropy loss treating the positive passage as the correct class among one positive and N negatives:

$$\mathcal{L}_{\text{ret}}(q, p^+) = -\log \frac{\exp(\text{sim}(q,p^+))}{\exp(\text{sim}(q,p^+)) + \sum_{j=1}^{N} \exp(\text{sim}(q,p_j^-))} \tag{3}$$

which encourages E_q and E_p to embed true pairs closer together than any negative pair [45]. After training, the retriever can generalize to new queries: it embeds the query and efficiently finds the nearest neighbor passages in the index. This retrieval step is crucial because it narrows down the evidence from potentially billions of tokens to a manageable subset that the generator will actually consider.

Generator Module (Conditional Seq2Seq Model). The generator in a RAG pipeline is typically a sequence-to-sequence language model that produces the final answer or output text, given the input query and retrieved documents. Formally, the generator defines a conditional distribution $P_{\text{gen}}(y \mid x, Z)$ over output sequences, where $Z = \{z_1, \ldots, z_K\}$ are the retrieved passages. The generator is often initialized from a pre-trained transformer-based seq2seq model (such as BART or T5) to leverage rich language generation capabilities [52]. During generation, the model is provided with the question (or prompt) as well as the content of the retrieved documents. There are multiple ways to feed the retrieved context to the generator:

- In an **early fusion** approach, one can concatenate the query x with the text of all retrieved passages into a single extended input sequence (perhaps with special separators) and have the seq2seq model attend to all of it at once. This is straightforward: the generator effectively treats the combined text as context and learns to pick out the relevant bits when producing the answer. However, this can be difficult if K is large, as the input may become very long.
- In the **late fusion** approach adopted by RAG [52], the generator considers one retrieved document at a time as a context and then marginalizes over the document choices (as in Eq. 1). Specifically, the RAG-Sequence variant fixes a single document z_i as context for generating the entire output and computes $P(y \mid x)$ by summing the probabilities $P(y \mid x, z_i)$ weighted by the retriever's confidence $P(z_i \mid x)$. Another variant, RAG-Token, allows the generator to switch between documents at the token level, effectively marginalizing over the document choice for each generated token [52]. In both cases, the generator $P_{\text{gen}}(y \mid x, z)$ itself works like a standard seq2seq model: it factorizes over the output tokens $y_1, \ldots, y_T$ as $\prod_{t=1}^{T} P_\theta(y_t \mid x, z, y_{<t})$, i.e. it generates one token at a time, attending to the input query x and the content of z (or multiple z's if early fusion). The generator's architecture typically uses an encoder-decoder Transformer: the encoder encodes the combination of x and z, and the decoder produces y autoregressively.

Because the generator is conditioning on retrieved evidence, it tends to produce outputs that are supported by that evidence. For example, if the query asks for a specific factual answer, the generator can copy or rephrase the needed information from one of the retrieved passages. This is in contrast to a vanilla language model which would attempt to rely on parametric knowledge (which might be outdated or incomplete). In sum, the generator module in RAG is responsible for fluent and coherent text generation, but crucially it is *grounded* by the retrieval context, which guides it toward accurate content.

Fusion Mechanisms and Answer Aggregation. A critical aspect of RAG systems is how to fuse information from multiple retrieved documents $z_1, \ldots, z_K$ when producing the final answer. Different fusion strategies have been explored: - **Marginalization (Probabilistic Fusion):** As described, RAG treats the retrieved documents as latent variables and marginalizes over them [52]. This means the model doesn't commit to one retrieved source up front; instead, it considers each in turn and combines their contributions by summing probabilities. Concretely, if y is an output sequence, a RAG model might compute its probability by $P(y|x) = \sum_{i=1}^{K} P_{\text{ret}}(z_i|x)\, P_{\text{gen}}(y|x, z_i)$ (Eq. 1). During training, this encourages the model to distribute probability mass across any

document that could yield the correct answer, reinforcing multiple evidence paths. At inference, one can approximately marginalize by taking the most likely y under this mixture model. - **Direct concatenation (Early Fusion):** As mentioned, another approach is to simply feed all top-K retrieved texts into the generator at once (often referred to as "Fusion-in-Decoder" when implemented in a decoder-attention context). In this setup, the generator effectively performs its own internal fusion by attending over a combined context. This approach has the advantage that the generator can directly cross-attend to multiple documents and integrate their content, but it may require a more powerful model to handle very long concatenated inputs. It also does not explicitly model the per-document probabilities $P(z_i|x)$. - Weighted Aggregation: Some systems introduce an attention or weighting mechanism over retrieved documents. For instance, the generator's decoder might assign different attention weights to different passages at each decoding step, effectively learning which source is most useful for generating the next token. This can be seen as a soft fusion: rather than hard marginalization or simple concatenation, the model dynamically blends information. In practice, approaches like [52] found that marginalization (which is a form of weighting by the retriever's scores) works well, especially when the retriever is accurate. Other works have since experimented with learnable fusion weights or iterative retrieval-generation cycles, but the core idea is the same: the model must reconcile possibly conflicting or complementary information from multiple documents to produce a single, coherent answer.

The choice of fusion affects the system's ability to handle conflicting evidence and the credit assignment during training (i.e., which document gets "credit" for a correct answer). RAG's probabilistic fusion provides a principled way to train the retriever and generator together by marginalizing, whereas direct concatenation treats the problem in a single forward pass of a generator (often fine for tasks where evidence is mostly additive or when using very large generators). Fusion strategies continue to be an active area of research, but they all serve the goal of effectively utilizing multiple retrieved pieces of text to improve answer completeness and correctness.

Training and Optimization. Training a RAG model involves objectives for both the retriever and the generator, which can be combined in an end-to-end manner. A common training approach is as follows: first, pre-train or initialize the retriever on a relevance task and initialize the generator on a language modeling or seq2seq task (often using a pre-trained model checkpoint). Then, perform joint fine-tuning on the target task (e.g., a QA dataset or a knowledge-intensive dialogue dataset) by maximizing the likelihood of the correct output y^* given the input x and allowing gradients to flow into both the generator and retriever. The training objective for the whole RAG system can be written as the expected negative log-likelihood:

$$\mathcal{L}_{\text{RAG}} = -\log P(y^* \mid x)\,,$$

where $P(y^* \mid x)$ is computed as in Eq. 1. Because $P(y^*|x)$ is a sum over documents, the gradient will encourage whichever retrieved documents z_i that helped predict y^* (by giving high $P_{\text{gen}}(y^*|x, z_i)$) to have their retrieval probability $P_{\text{ret}}(z_i|x)$ increased. In effect, the model learns to adjust the retriever to fetch better supporting documents and adjust the generator to rely on them appropriately. This joint training is typically done with standard backpropagation; since the retriever's selection operation is not differentiable for all documents, one uses the top-K approximation (only those contribute to the loss) and treats the retrieval probabilities for those as soft variables. [52] report that initializing the retriever with DPR and then fine-tuning end-to-end yields the best results, as opposed to training from scratch. Notably, the retriever is trained indirectly here: it does not receive explicit labels of which document is correct, but the generator's success or failure on producing y^* provides a supervision signal. This is sometimes called "self-supervised" retriever training or "feedback" training.

In addition to end-to-end training, various optimization tricks may be used: e.g., using a small learning rate for the retriever if it's already strong, or alternating between retriever-focused and generator-focused updates. In some cases, researchers have also explored **contrastive learning at the generation level** (to reduce ambiguity between retrieved passages) or reward-based objectives if the task is not a straightforward next-word prediction. However, the most common training objective for RAG is the simple maximum likelihood training of the seq2seq model, augmented by the latent document marginalization. The result is a system where both components are tuned to the end task: the retriever learns to bring useful evidence, and the generator learns to incorporate that evidence into the output. This joint optimization is a major advantage of RAG over non-integrated pipelines, as it

aligns the retriever's objective with generating correct final answers (not just retrieving vaguely related documents).

3.3 Historical Context

The evolution of RAG builds upon earlier developments in open-domain question answering (QA) and neural information retrieval. Traditional open-domain QA systems were typically pipeline-based, consisting of a retrieval step followed by a reading or extraction step [12]. For example, [12] introduced the DrQA system, which first used a TF-IDF or BM25 retriever to select Wikipedia articles and then fed those to a machine reader model to extract answers. This established the value of retrieving relevant text from a large corpus as an essential first step in answering open-domain questions. However, in such pipeline approaches the retriever was not integrated into the learning of the reader, and the system could not adjust retrieval based on the end task's needs. Subsequent research sought to bridge this gap by jointly learning retrieval and answering. Notably, the concept of using learned dense representations for retrieval emerged as a powerful alternative to traditional sparse retrieval. Early milestones in this direction include *latent retrieval* models: [48] proposed the ORQA model, which treats retrieval as a latent variable problem and pre-trains a neural retriever on an unsupervised "inverse cloze task" before jointly fine-tuning it with a reader on QA. Around the same time, [26] introduced REALM, a retrieval-augmented language model pre-training method that incorporated a differentiable retriever into the pre-training of a masked language model. REALM demonstrated that pre-training a model to retrieve and reason over Wikipedia could significantly improve open-domain QA, highlighting the benefit of coupling a language model with a learned retrieval mechanism [26]. These efforts were focused primarily on question answering (often extractive), but they laid important groundwork for retrieval-augmented generation by showing that retrieval and neural text generation can be trained in tandem.

In parallel, the idea of combining external knowledge with neural networks has roots in earlier **memory-augmented models**. For instance, Memory Networks [95] and subsequent variants allowed neural networks to read from an external memory of facts and use that information to answer questions or generate responses. These models (e.g., [95, 87]) demonstrated the feasibility of non-parametric memory for reasoning, albeit on smaller-scale knowledge bases or synthetic tasks. While memory network architectures were often task-specific and required the memory to be relatively small or structured, they presaged the RAG approach by emphasizing that not all knowledge needs to be baked into model parameters—some can be looked up as needed. Another line of work in dialogue systems also integrated retrieval into generation: the **Wizard of Wikipedia** project [18] is a prime example, where a conversational agent retrieves relevant Wikipedia sentences and conditions a generative dialogue model on those sentences to produce knowledgeable responses. This retrieval-based dialogue system (published in 2019) demonstrated improved factuality and depth in conversational responses, reflecting the general trend that augmenting generators with retrieved context yields more informative and correct outputs.

These developments converged in 2020 with the formalization of Retrieval-Augmented Generation by [52], who unified the retriever-reader architecture with seq2seq generation in an end-to-end framework. The RAG model of [52] was a culmination of insights from open-domain QA and neural IR: it used a **dense passage retriever** [45] to fetch text chunks from Wikipedia and a powerful seq2seq generator (BART) to produce answers or summaries, training both components jointly. By marginalizing over multiple retrieved documents (as in Eq. 1), the RAG system could leverage several pieces of evidence and was shown to outperform both parametric-only models and earlier retrieve-and-read pipelines on knowledge-intensive tasks like open QA [52]. The introduction of RAG in 2020 is considered a key milestone because it generalized retrieval-augmented architectures beyond QA to any generative task requiring external knowledge. It also spurred a new line of research into **knowledge-enhanced text generation**, influencing subsequent models that further refined retrieval modules, document ranking, and fusion techniques for even better performance.

Execution Flow of a RAG System. To summarize the interactions of these components, we outline the step-by-step execution flow of a typical RAG system processing a query:

1. **Query Encoding:** Given an input query x (e.g., a question in natural language), the retriever's query encoder E_q first encodes x into a vector representation v_q. This vector captures the semantic meaning of the query in a dense

embedding space.

2. **Document Retrieval:** Using the query vector v_q, the system performs a search over the document index (the external corpus $\mathcal{C}$). It computes similarity scores $\text{sim}(x, d)$ for documents d (often via inner product with pre-computed document embeddings $E_p(d)$) and retrieves the top K documents with highest scores. These top-K documents $Z = \{z_1, ..., z_K\}$ are assumed to be the most relevant pieces of text related to the query.

3. **Context Preparation:** The text of the retrieved documents Z is retrieved from the knowledge store. The RAG system now has access to these K passages (e.g., paragraphs from Wikipedia) that can serve as supporting context. Depending on the fusion strategy, the system either concatenates these passages or will handle them separately in the next step.

4. **Answer Generation:** The query x along with the retrieved context Z are fed into the generator model. If using early fusion, all of Z (or a subset, if some cutoff like K' is used) is given as additional input to the encoder. If using late fusion (as in the original RAG), the generator will consider each z_i in Z in turn. The generator's decoder then produces an output sequence y (the answer or response). During this decoding, the model may attend to relevant parts of the retrieved texts. For example, the decoder might focus on a specific retrieved passage that contains a needed fact when generating the corresponding part of the answer. In RAG's late fusion, the decoder actually generates an answer for each z_i implicitly and the probabilities of tokens are combined by marginalization. In practice, one can sample or beam-search for the best output y across the combined evidence.

5. **Fusion and Output:** If multiple candidate outputs were considered (e.g., one per retrieved document), the model marginalizes or otherwise aggregates them to produce the final answer. Often the single most likely sequence y is selected as the output. The final answer y is then returned by the system as the response to the query. Optionally, the system might also output the passages it used (providing provenance or justification, which is a useful feature of RAG systems).

This flow involves interleaving retrieval and generation in a seamless way. Notably, steps 1–2 (retrieval) drastically reduce the problem space by focusing on a handful of documents out of potentially millions, and steps 3–5 ensure that the information in those documents is synthesized into a fluent answer. The entire process is typically very fast at inference: encoding the query and searching the index can be done in tens of milliseconds with efficient vector databases, and generation is on the order of the length of the output (with modern transformers generating dozens of tokens per second). Therefore, RAG systems can scale to handle user queries in real-time applications, all while maintaining higher accuracy by leveraging updated and explicit knowledge. The end-to-end design means that if the output is incorrect, the system can be improved by either enhancing the retriever (fetch more relevant docs) or the generator (better use of docs), or both, which aligns with the modular evaluation and training typical in IR+NLP pipeline but now integrated into a single model.

4 Year-by-Year Progress in RAG

RAG, as it is known today, was proposed in [52], but before then, the retrieval and read pipeline, which operates like RAG. This section reviews the evolution of RAG from the pre RAG erra till date. The anual year-by-year progress of RAG is illustrated in Figure 2.

4.1 Initial Proposals and Early Research (2017–2019)

Before the term *retrieval-augmented generation (RAG)* was coined, researchers explored methods to combine information retrieval with neural models for question answering (QA) and text generation. Early open-domain QA systems typically employed a *retrieve-and-read* pipeline: a search module locates relevant documents, then a neural reader model extracts or generates answers [12]. For instance, the 2017 *DrQA* framework answered questions using Wikipedia as a knowledge source by pairing a TF-IDF-based document retriever with an RNN-based reader trained to extract answer spans. Although [12] demonstrated strong performance on open-domain trivia tasks, these systems were piecemeal in nature: the retrieval and

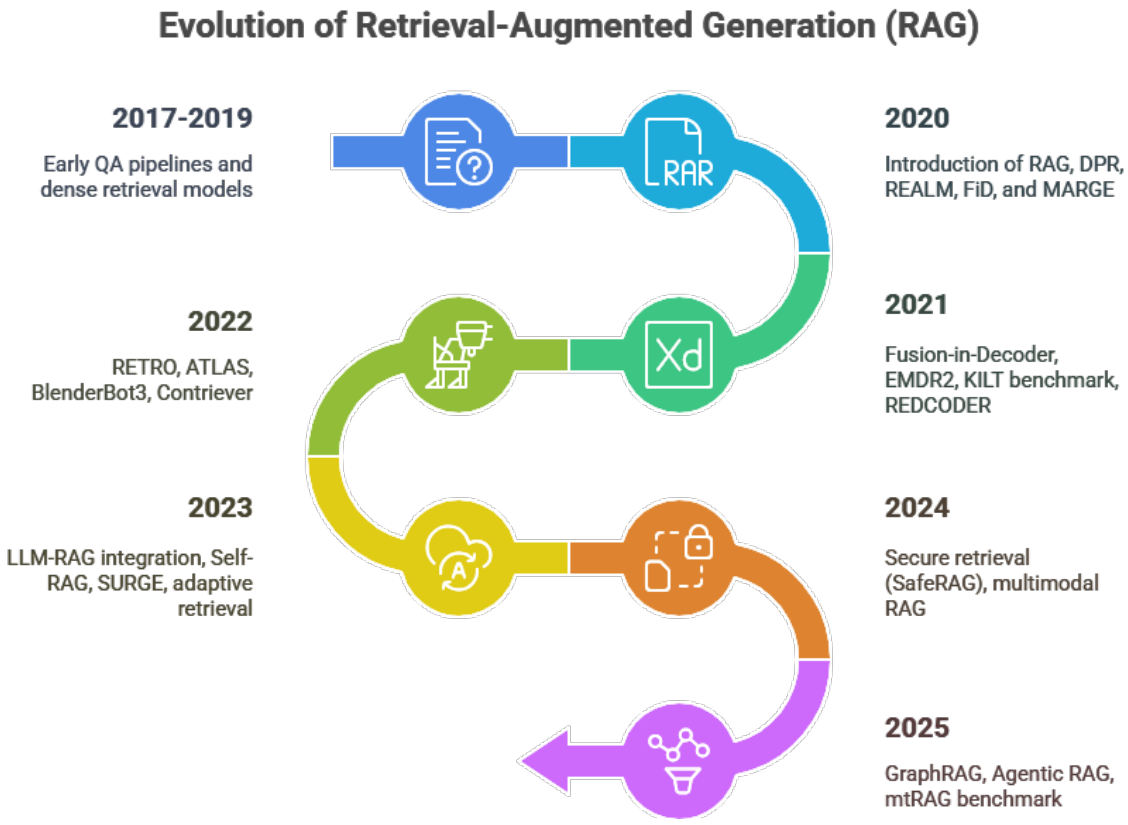

Figure 2: Evolution of a RAG Architecture.

generation components were not trained jointly, and the end-to-end approach was limited to extractive answers. In 2018, work shifted toward tighter integration between retrieval and reading. [92] proposed R^3 (*Reinforced Reader-Ranker*), adding a neural ranker to score retrieved passages by answer likelihood. The system then learned ranker–reader synergy via reinforcement learning, boosting open-domain QA accuracy by better filtering relevant evidence. Meanwhile, neural IR methods gained traction. [48] introduced *Latent Retrieval* in their *Open-Retrieval QA* (ORQA) framework, training a *dense* retriever and a reader end-to-end with only question–answer supervision. Specifically, the retriever embeddings were pretrained on an inverse cloze task and then adapted to select evidence documents that help the QA model answer correctly. This concept of *dense retrieval*, which outperformed sparse BM25 by up to 19% in exact-match QA scores, would become the foundation of subsequent RAG models. Still, these early systems were limited mostly to extractive QA, without a unified end-to-end training for generative outputs.

4.2 Major Milestones (2020–2024)

2020 — Birth of RAG. The year 2020 marked a turning point with the official formalization of *retrieval-augmented generation*. [52] coined the term "RAG" and demonstrated its power on knowledge-intensive tasks. RAG explicitly splits knowledge across (i) a neural *retriever* and (ii) a neural *generator*, each playing a distinct role. Given a query x, RAG retrieves top-k relevant passages $\{z_1, \ldots, z_k\}$ from a large text corpus (via a learned dense index) and then conditions a seq2seq model on both x and $\{z_i\}$. Mathematically,

$$P(y \mid x) \;=\; \sum_{i=1}^{k} P_\theta(y \mid x, z_i)\; P_\phi(z_i \mid x),$$

where $P_\phi(z_i \mid x)$ is the retriever's distribution over documents (often realized via a softmax on

inner product scores), and $P_\theta(y \mid x, z_i)$ is the conditional probability of the generator. [52] used a BART-based generator and a Dense Passage Retriever [45], outperforming older retrieve-and-read pipelines by leveraging both parametric and non-parametric memory. Simultaneously, [26] introduced *REALM*: retrieval-augmented *pretraining*,

which integrated a differentiable retriever into a language model to predict masked tokens with retrieved evidence. REALM achieved significant QA gains over conventional LMs, validating that external knowledge injection helps both at pretraining and fine-tuning. [45] also released their Dense Passage Retrieval (DPR) approach, establishing a simple but effective dual-encoder architecture. By 2020's end, retrieval-augmented strategies had become key to state-of-the-art QA, and generative models demonstrated improved factuality by referencing retrieved text. Some particular progress made in 2020 are discussed below:

Dense neural retrieval. Classical TF–IDF/BM25 retrieval limited earlier pipeline QA systems. [44] introduced *Dense Passage Retrieval (DPR)*, training dual BERT encoders to embed questions and passages into a shared space. DPR improved top 20 recall by 9–19 pp over BM25 and became the de facto index for RAG models.

End-to-end retriever–generator architectures. Building on DPR, [51] proposed the eponymous *RAG* model: a BART generator conditions on k passages retrieved (and jointly trained) via DPR. Two variants—RAG-Sequence and RAG-Token—achieved state-of-the-art exact-match scores on Natural Questions and TriviaQA, while producing more specific, better-grounded answers than closed-book T5 of comparable size. Concurrently, the *Fusion-in-Decoder (FiD)* architecture of [34] showed that a T5 decoder attending over dozens of retrieved passages could push QA accuracy even higher, underscoring that modern seq2seq models can synthesise evidence from many documents.

Retrieval-aware pre-training. Rather than bolt retrieval on during fine-tuning, REALM [25] trained a BERT-style model to retrieve Wikipedia passages to fill masked tokens, optimising retriever and LM jointly. REALM outperformed larger closed-book models by up to 16 pp in open QA. MARGE [49] extended the idea: the model reconstructs a target document from related texts it retrieves, yielding strong zero-shot results in multilingual summarisation and translation.

Applications beyond QA. RAG principles generalized quickly. In knowledge-grounded dialogue, bridging a prior (inference-time) and posterior (training-time) retriever improved response relevance [15]. Fact-checking systems retrieved evidence then generated verdicts, while early experiments hinted at factuality gains in abstractive summarisation. The *KILT* benchmark [68] unified eleven knowledge-intensive tasks with a shared Wikipedia snapshot and evaluation that scores both answer correctness and evidence retrieval. A single RAG baseline proved competitive across QA, dialogue, fact verification, and slot-filling, highlighting RAG's domain-agnostic promise.

Impact and open questions. By year-end 2020, RAG systems of only a few hundred million parameters were surpassing 11-billion-parameter closed-book LMs [52, 75], demonstrating the efficiency of hybrid parametric + non-parametric memory. Challenges that remained—in retrieval latency at scale, multi-hop reasoning, and tighter faithfulness evaluation—set the agenda for subsequent work. Nonetheless, the 2020 breakthroughs firmly established retrieval-augmented generation as a core methodology for building knowledgeable, transparent, and updatable NLP systems.

2021 — Improving Generation Quality. Work in 2021 centered on refining how retrieval and generation interact, leading to *Fusion-in-Decoder* (*FiD*) by [34]. Rather than marginalize over top-k passages as in RAG, FiD concatenates them and feeds them all into a T5-based seq2seq, allowing the decoder to attend to multiple documents simultaneously. This architecture demonstrated that large generative models can effectively combine evidence from many passages, achieving further gains on open QA benchmarks. Meanwhile, new tasks beyond QA surfaced, such as fact-checking, knowledge-grounded dialogue, and entity-rich tasks [68], all relying on retrieval to supply correct and up-to-date information. By 2021, RAG had expanded its reach to a wider range of knowledge-intensive NLP domains. Some major progress made in 2021 are discussed below:

Open-domain QA was an early focus of RAG research. [34] introduced the Fusion-in-Decoder (FiD) architecture, which encodes each retrieved passage independently and fuses them in the decoder. FiD achieved state-of-the-art results on benchmarks such as Natural Questions and TriviaQA, showing that performance improves as more evidence passages are. In parallel, [76] developed an end-to-end training scheme (EMDR2) for multi-document QA. By treating retrieval as a latent variable and iteratively training the retriever and reader with an

EM-like algorithm, they improved answer accuracy across datasets, establishing new state-of-the-art results without supervised retrieval labelsyielded. Together, these works demonstrated that advanced RAG architectures and training strategies yield substantial QA gains in 2021.

RAG techniques also permeated knowledge-grounded dialogue system. The KILT benchmark introduced by [69] unified several knowledge-intensive tasks (QA, fact-checking, dialogue, etc.) on a common Wikipedia knowledge source. KILT showed that a shared dense retriever and generative model can serve as a strong generalist system: it outperformed task-specific baselines in fact-checking and open-domain QA, and was competitive on dialogue and entity linking:contentReference[:4]index=4. This suggests that RAG-style models (a retriever plus a seq2seq generator) can effectively support multi-turn, knowledge-grounded conversation. While Wizard-of-Wikipedia (2019) had earlier demonstrated retrieval in dialogue, the KILT results in 2021 underscored that unified RAG pipelines are robust across dialogue and QA domains.

In summarization and content generation, RAG has been applied to integrate relevant context. One example is code summarization: [67] proposed REDCODER, a framework that retrieves relevant code snippets or summaries to augment a code generation model. By searching a codebase for similar examples, REDCODER improved both code generation and summarization quality on Java and Python benchmarks. More broadly, retrieval-augmented summarization was noted to help produce more accurate and up-to-date summaries by grounding language models in external documents (e.g. news or scientific articles), though in 2021 most demonstrations were in specialized domains like code or clinical summarization. Fact-checking similarly benefited: retrieving evidence from text (e.g. news or scientific literature) provides the basis for verifying model outputs. KILT explicitly includes fact-checking (the FEVER dataset) in its suite, and found that a general RAG approach is competitive on evidence retrieval for claims.

Several methodological improvements emerged in 2021. Beyond FiD and EMDR2, researchers explored how to structure RAG models. [51] distinguished two RAG variants: one conditions on a fixed set of retrieved passages for the entire output, while the other can fetch different passages at each decoding step. [34] exemplified the former by concatenating encoded passages in the decoder. but in 2021 the focus was on multi-passage integration. Additionally, enhancements to retrievers were important: dense retrievers like DPR [44] underpin many 2021 RAG systems, often pretrained on open-domain QA. Training retrievers jointly with generators, as in [77], improved retrieval relevance. Some studies also integrated topic or dialogue context into retrieval for conversational settings. In all cases, the synergy of retrieval and generation architectures was a key theme.

By the end of 2021, standardized benchmarks and code releases helped consolidate RAG progress. The KILT benchmark in particular brought together 11 tasks across multiple knowledge-intensive domains. On this benchmark, the combination of a pretrained retriever and a large generative model achieved strong performance across QA, dialogue, and even slot filling, highlighting the versatility of RAG pipelines. In open QA, the Natural Questions, TriviaQA, and EfficientQA datasets continued to serve as metrics for RAG-based methods; FiD and other models raised the bar on these tasks in 2021. Summaries of RAG results noted major gains over prior baselines in both accuracy and factuality.

2022 — Scaling & Specialization. In 2022, [10] introduced *RETRO* (*Retrieval-Enhanced Transformer*), revealing that a moderately sized 7.5B-parameter model could match GPT-3 (175B params) if it retrieves relevant text chunks from a huge corpus (2T tokens), thereby proving that retrieval can *substitute for scale*. RETRO thus illustrated the efficiency advantage of external knowledge over solely increasing model parameters. In parallel, [35] presented *Atlas*, a retrieval-augmented language model aiming for *few-shot learning* on knowledge-intensive tasks. Atlas combined a T5-based generator with an advanced dense retriever, pre-trained on massive unlabeled text. Demonstrating strong results in few-shot scenarios (only 64 examples needed for respectable performance), Atlas highlighted the synergy between retrieval and pre-trained seq2seq in reducing reliance on large labeled datasets.

[52] and DPR [44] also proved that conditioning generation on retrieved passages boosts factual accuracy. DeepMind's **RETRO** model [9] attaches a nearest-neighbour retriever to every decoding window of a 7-billion-parameter transformer; with a 2-trillion-token index, RETRO matches GPT-3

performance while using 25× fewer parameters, underscoring retrieval's efficiency gains.

Google's **ATLAS** [35] unifies retrieval and generation during *pre-training*. Using the unsupervised **Contriever** dense retriever [37] and a Fusion-in-Decoder reader, ATLAS achieves new state-of-the-art results on Natural Questions and TriviaQA and, in few-shot mode, outperforms much larger PaLM-540B by 3 EM on NQ with only 64 examples. These results highlight that high-quality retrieval plus multi-document fusion can outperform sheer parameter count.

Beyond QA, RAG became core to knowledge-grounded *dialogue*. Meta's **BlenderBot 3** [80] couples a 175B LLaMA-style generator with live internet search and long-term memory, reducing hallucinations and increasing user engagement in open-domain conversation. BlenderBot 3's deployment study shows that users prefer retrieval-grounded responses and that continual online learning can keep the retriever index current. Retrieval quality itself improved through unsupervised contrastive training. Contriever [37] dispenses with labelled (question, passage) pairs, yet surpasses BM25 and even DPR on BEIR benchmarks, making high-recall indexes available for any corpus. Such retrievers power ATLAS and other 2022 RAG systems, demonstrating that scalable training data is no longer a bottleneck.

Finally, 2022 work extended RAG to *fact-checking, summarization, and few-shot learning*. ATLAS gains 5F1 on FEVER, indicating that retrieved evidence helps verdict generation. Early studies in retrieval-augmented summarization show improved factual consistency by grounding summaries in external documents. Few-shot evaluations reveal that retrieval narrows the data gap: ATLAS and RETRO deliver strong accuracy with under 100 task examples, whereas closed-book baselines require orders of magnitude more data.

Outlook By the end of 2022, RAG had broadened from open-domain QA into a general recipe for knowledge-intensive NLP. Parameter-efficient hybrids like RETRO and ATLAS challenge the notion that bigger models alone yield better knowledge; instead, high-quality retrieval and multi-document reasoning emerge as key levers. Open challenges include faster retrieval over trillion-token corpora, differentiable multi-hop reasoning, and robust evaluation of evidential faithfulness, but 2022 firmly established retrieval-augmented generation as a premier path toward up-to-date, factual, and data-efficient language models.

2023 — RAG Meets LLMs. By 2023, mainstream LLM-based applications (e.g., ChatGPT with plugins, Bing Chat, enterprise chatbots) widely incorporated retrieval [27]. This *retrieve-then-generate* paradigm was used to mitigate *hallucinations* and update factual knowledge post-training. The debate emerged as to whether *long-context LLMs* (with tens of thousands of tokens) could negate the need for retrieval systems. Studies like [54] showed that while large context windows can absorb more text, RAG remains more *cost-efficient* and better at exposing citations. Hybrid approaches also arose: letting a model choose between retrieving or just using a long context. Overall, RAG became a cornerstone for credible LLM deployments needing up-to-date knowledge and interpretability. SOme of the areas that received attention in 2023 are discussed below:

Scale and few-shot learning: Atlas[36], an 11B-parameter retrieval-augmented model, achieved 42.4% accuracy on Natural Questions with only 64 training examples, outperforming a 540B closed-book model by 3%. Atlas also set new few-shot records on TriviaQA and FEVER (gains of +3–5%), matching 540B-scale performance on multi-task benchmarks. Crucially, Atlas's dense document index can be easily updated with new text, demonstrating updatable knowledge.

Adaptive retrieval: Self-RAG[5] trains a single language model to generate special "reflection" tokens that trigger on-demand retrieval and self-critique. In experiments, 7B and 13B Self-RAG models substantially outperformed ChatGPT and a RAG-augmented Llama-2-chat baseline on open-domain QA, reasoning, and fact-verification tasks, yielding much higher factual accuracy and citation precision.

Knowledge-grounded dialogue: Retrieval augments dialogue systems to improve consistency and informativeness. Kumari et al.[47] incorporate retrieved persona and context snippets in long conversation modeling, showing that adding relevant knowledge improves response quality. Similarly, Kang et al.[43] propose *SURGE*, which retrieves relevant subgraphs from a knowledge graph and uses them to bias the response generation. SURGE produces more coherent, factual responses grounded in the retrieved

knowledge.

Summarization and explanation: RAG has been applied to summarization and explanation tasks. By retrieving source documents or evidence passages, RAG-augmented summarizers produce more accurate and detailed summaries than closed-book models. Likewise, in fact-checking pipelines, retrieving evidence before verification leads to more reliable verdicts and explanations. These applications extend RAG's grounding advantages beyond QA to a broader range of generative tasks.

Benchmarks and evaluation: Knowledge-intensive benchmarks track RAG progress. OpenQA tasks (Natural Questions, TriviaQA, HotpotQA) and the KILT benchmark suite (including QA, fact-checking, slot filling, etc.) are standard evaluation sets. In 2023, RAG models dominated many KILT tasks and few-shot QA challenges. New evaluation tools also emerged: for example, the RAGAS framework provides reference-free metrics for RAG pipelines, and the RAGTruth corpus (Niu et al. 2024) enables fine-grained analysis of hallucinations in RAG outputs. 2023 also witnessed a major spike in publications and adoption of RAG. Some other works are on Active RAG [38], Improving domain adaptation of RAG models for open question answering [85], content filtering for RAG [94], and RAG with self-memory [16].

2024 — Recent Advances. In 2024, research on RAG continues to push on *secure retrieval* frameworks, multi-hop reasoning, and domain specialization. Several groups explore *differentiable retrievers* that can be tuned in an end-to-end pipeline, while others investigate merging large-context attention with retrieval indexing. Meanwhile, new techniques aim to reduce the chance the generator ignores retrieved data or merges contradictory documents incorrectly. RAG-based chat systems in healthcare, finance, and law now incorporate advanced fact-checking modules to ensure that only *vetted* external sources influence the final output. Some notable works in 2024 are Evaluating Retrieval Quality in RAG [78], Benchmarking RAG for Medicine [97], Benchmarking LLM in RAG [13], Review of RAG for AI Generated Content [106], Unifying Context Ranking with RAG [100], Searching for the Best Practice in RAG [93], Finetuning Vs RAG for Less Popular Knowledge [86], Integrating RAG wit LLM in Nephrology [61], RAG for Copyright Protection [23], RAG for Textual Graph Understanding and Question Answering [31], Interactive AI with RAG for NExt Generation Networking [103], Web Application for RAG: Implementation and Testing [72], Overcoming Challenges for Long Input in RAG [39], and Adapting Language Model for Domain Specific RAG [104]. Some specifics in RAG development as at 2024 are discussed below:

Infrastructure and standardized evaluation: The community recognised a need for common tooling and shared tasks. *Ragnarök* introduced a reusable end-to-end framework and provided industrial baselines for the inaugural **TREC 2024 RAG Track** [70]. Beyond code, evaluation methodology itself became a research focus: **ARAGOG** proposed automatic grading of RAG outputs that correlates with human judgements, analysing retrieval precision and answer similarity across advanced pipelines [20]. These efforts mark a shift from anecdotal demos to systematic, reproducible assessment.

Adaptive retrieval and self-reflection: Building on ideas such as Self-RAG, several 2024 works taught models to *decide when—and how much—to retrieve*. SAM-RAG dynamically filters documents and verifies both evidence and final answers in multimodal contexts, improving accuracy without unnecessary retrieval calls [102]. For complex visual-question-answering, *OmniSearch* plans multi-hop retrieval chains on the fly, demonstrating large gains on the new Dyn-VQA benchmark [53]. These results confirm that retrieval policies, not just retriever quality, matter for difficult queries.

Multimodal RAG breaks out: Where earlier RAG research was text-only, 2024 saw a surge in multimodal extensions. SAM-RAG and OmniSearch both combine text and image evidence, while concurrent frameworks (e.g. mR^{2}AG and M3DocRAG) introduce retrieval–reflection loops or structured vision–language indexes. Surveys published this year chart the design space and highlight open issues such as cross-modal alignment and vision-aware reranking [102, 53].

Progress in dense retrieval: High-recall retrievers remain the backbone of every RAG system. 2024 research emphasised *unsupervised or instruction-tuned* retrievers that avoid costly labelled data, building on contrastive pre-training techniques and LLM-augmented embedding models. These retrievers power the top submissions in the TREC track and underpin production deployments discussed in industrial white papers.

Rapid growth and forward outlook: A bibliometric snapshot counted more than 1,200 RAG-related papers on arXiv in 2024 alone [106],

compared with fewer than 100 the previous year, underscoring the field's rapid maturation. Looking ahead, challenges include ultra-fast retrieval over trillion-token corpora, faithfulness verification for multi-hop reasoning, and energy-efficient multimodal indexing. Nevertheless, 2024 firmly established RAG as the default strategy for grounding large language (and vision-language) models in up-to-date, attributable knowledge.

4.3 2025 — The Current Direction

The community is exploring how to marry graph knowledge with text retrieval. A comprehensive survey formalised the *GraphRAG* paradigm and mapped design choices for graph-aware retrievers and generators [29]. A companion study compared vanilla RAG and GraphRAG across QA and summarisation, showing complementary strengths and proposing hybrid fusion strategies [28].

Security and robustness. *SafeRAG* introduced the first security benchmark for RAG pipelines, cataloguing four attack classes (silver noise, context conflict, soft-ad, DoS) and demonstrating that 14 representative systems fail even simple manipulations [55]. Findings sparked interest in provenance tracking and adversarially trained retrievers.

Agentic and selective retrieval. A survey on *Agentic RAG* synthesised emerging patterns—reflection, planning, tool use—and argued that autonomous agents can orchestrate multi-hop retrieval more effectively than static pipelines [83]. Concrete instantiations followed:

- *SIM-RAG* learns a self-skeptic critic that decides when to stop multi-round search, cutting redundant calls and boosting exact-match on five QA sets by up to 4pp[98].
- *Self-Routing RAG* jointly trains an LLM to choose between internal knowledge and external retrieval, reducing retrieval volume 29% while raising accuracy 5pp[96].

Conversational evaluation at scale. IBM's *mtRAG* benchmark is filling a gap in multi-turn assessment: 110 human-written conversations (7.7 turns each) across four domains, plus synthetic variants, revealed that state-of-the-art systems struggle with unanswerable and follow-up questions citep katsis2025mtrag. Alongside automatic judges such as RL-F and RB-LLM, mtRAG enables holistic measurement of retrieval, generation, and turn-level faithfulness.

Emerging directions. Workshops and surveys highlighted three open fronts. First, *agentic planning*—letting models reason over retrieval actions—promises better long-horizon reasoning but raises cost and safety questions. Second, *structured retrieval* (graphs, tables, multimodal stores) demands new embedding spaces and fusion operators. Third, *secure and privacy-preserving RAG* is gaining urgency, with SafeRAG prompting work on attack-aware retrievers and watermarking of retrieved evidence.

Outlook Mid-2025 results suggest that modest-sized, security-hardened, agent-controlled RAG systems can rival much larger closed-book LMs while offering provenance. Key challenges ahead include trusted evidence ranking at trillion-document scale, automatic detection of retrieval-based attacks, and seamless integration of non-text modalities. Nevertheless, 2025 firmly positions RAG not merely as a booster of accuracy but as an essential framework for *reliable*, *updatable*, and *auditable* language agents.

5 RAG for Proprietary Data - Industry Implementation

5.1 Current Approaches

Organizations increasingly apply RAG to *private internal knowledge*, using a secure pipeline to retrieve from proprietary documents and feed them to a generative model. This often involves *on-premise or VPC-hosted* vector databases, ensuring that queries and document embeddings never leave the corporate firewall [60, 33]. Enterprises store their text in an index, and at inference time, a local embedding model transforms a user query into a dense vector to find top-k relevant chunks. These chunks are appended to the user query as context for an LLM (e.g., GPT-4, or a self-hosted model). The separation of knowledge in a database reduces the risk that proprietary data leaks into the model's parameters, but it does not fully guarantee privacy—the generator might still reveal sensitive details in the output. To mitigate these risks, some enterprise solutions incorporate *access control layers* that filter out documents the user lacks permission to see. Others experiment with *secure enclaves* or homomorphic encryption to blind the retrieval operation [105]. However, performance overhead is non-trivial. Another strategy is to *fine-tune* an LLM on domain data. Although fine-tuning helps the model internalize domain nuances, it can potentially memorize private text. RAG better suits *data freshness*: the knowledge store can be updated without retraining.

5.2 Industry Implementation Examples

In recent years, several organizations have explored the deployment of Retrieval-Augmented Generation (RAG) systems to leverage proprietary data effectively. Notable case studies include:

PGA Tour's Use of RAG for Enhanced Information Retrieval The PGA Tour implemented a RAG system to improve information retrieval related to golf events and player statistics. By integrating their extensive proprietary data into the RAG framework, they enabled more accurate and contextually relevant responses to user queries. This approach addressed previous limitations where general AI models lacked specific domain knowledge, leading to inaccuracies in responses. The RAG system allowed the PGA Tour to provide precise information, enhancing user engagement and satisfaction [11].

Bayer's Application of RAG in Agricultural Data Management Bayer utilized RAG to manage and retrieve proprietary agricultural data, aiming to provide farmers with accurate and timely information. By incorporating their extensive datasets into the RAG system, Bayer enhanced the accessibility and usability of critical agricultural information. This integration facilitated better decision-making processes for farmers, leading to improved crop management and productivity [11].

Rocket Companies' Integration of RAG for Mortgage Processing Rocket Companies explored the use of RAG to streamline mortgage processing by integrating proprietary financial data. The RAG system enabled more efficient retrieval of relevant information, reducing processing times and improving customer experiences. By leveraging their internal data within the RAG framework, Rocket Companies enhanced the accuracy and speed of their services, demonstrating the potential of RAG in financial applications [11].

Shorenstein Properties' Implementation of RAG for Data Organization Shorenstein Properties adopted RAG to automate file tagging and organize proprietary data more efficiently. By integrating their internal documents into the RAG system, they improved data accessibility and management. This implementation showcased RAG's capability to handle complex data organization tasks, leading to increased operational efficiency within the company [11].

Cohere's Advancement of RAG for Source Citation Cohere advanced RAG technology to ensure AI systems cite their sources, facilitating human verification of the information produced. By integrating external texts such as company documents or news websites, Cohere's RAG system reduced errors known as hallucinations and provided access to current information. This development highlighted RAG's potential in enhancing the reliability and transparency of AI-generated content [50].

NVIDIA's Deployment of RAG in Enterprise AI Solutions NVIDIA incorporated RAG into their enterprise AI solutions to connect large language models with specific information, such as proprietary customer data and authoritative research. This integration enabled more accurate and relevant responses to user queries, enhancing productivity and protecting against AI hallucinations. NVIDIA's deployment demonstrated RAG's applicability in various industries, including customer service and data management [65].

IBM's Utilization of RAG for Domain-Specific AI Applications IBM employed RAG to equip models with specific information, such as proprietary customer data and authoritative research documents. This approach allowed their AI systems to incorporate up-to-date information into generated responses, improving accuracy and relevance in domain-specific applications. IBM's utilization of RAG showcased its effectiveness in enhancing AI capabilities across different sectors [32].

These case studies illustrate the diverse applications of RAG in leveraging proprietary data across various industries. By integrating internal datasets into RAG systems, organizations have enhanced information retrieval, improved decision-making processes, and increased operational efficiency. However, challenges such as data privacy, intellectual property concerns, and computational overhead remain areas for further research and development.

5.3 Emergent Trends and Patterns in Industry Implementation

Accuracy vs. Latency Trade-offs: Models optimized for retrieval accuracy (e.g., FiD, WebGPT) typically have higher latency due to multiple document processing. Industry applications prioritize real-time performance (e.g., NVIDIA RAG aims for sub-second latency).

Scaling Beyond Wikipedia: Early RAG models focused on Wikipedia-scale retrieval, while industry applications integrate proprietary databases and real-time web data. WebGPT and IBM Watsonx RAG retrieve from dynamic knowledge bases, offering more up-to-date responses.

Reducing Hallucination: There is a shift toward factual consistency. WebGPT and IBM Watsonx RAG enforce citation mechanisms to enhance factual accuracy.

Model Size vs. Retrieval Efficiency: Smaller models augmented with retrieval (e.g., Atlas, NVIDIA RAG) can outperform much larger models without retrieval (e.g., GPT-3 175B), demonstrating that knowledge retrieval reduces the need for extreme model scaling.

Enterprise Adaptation: Academic models optimize for benchmark performance, whereas enterprise RAG systems prioritize real-world reliability, integration with databases, and data privacy compliance.

6 Evaluation of RAG Systems

The evaluation of RAG systems is multifaceted, as performance depends not only on the generative model but also on the quality of the retrieval pipeline. A robust evaluation framework must assess retrieval accuracy, answer quality, factuality, latency, and scalability. This section provides a structured overview of RAG evaluation criteria, benchmarks, and the impact of architectural design choices. Summary of this section is in Table Table 1

6.1 Evaluation Dimensions and Metrics

RAG performance is commonly assessed along the following dimensions:

Retrieval Accuracy. Key metrics include:

- **Recall@k** – Measures the proportion of queries where a relevant document appears among the top-k retrieved results.
- **Mean Reciprocal Rank (MRR)** – Captures the average inverse rank of the first relevant document, rewarding early placement.
- **Mean Average Precision (MAP)** – Evaluates the quality of ranked retrieval across relevant items.

Generation Quality. Evaluated using:

- **Exact Match (EM)** and **F1 Score** – Common in QA tasks to measure overlap with reference answers.
- **BLEU**, **ROUGE** – N-gram-based measures used in summarization and long-form generation.
- **Faithfulness / Hallucination Rate** – Human or automated evaluations of factual consistency with retrieved sources.

Efficiency and Latency. These include:

- **Retrieval time**, **generation latency**, and **end-to-end response time**.
- **Memory and compute requirements** – Especially important for deploying RAG systems at scale.

Scalability. As corpus size grows, the system's ability to maintain retrieval quality and generation fidelity is tested. Evaluation considers:

- **Index size vs. retrieval accuracy**.
- **Adaptability to new or evolving data without retraining**.

6.2 Benchmarks and Datasets

Several benchmarks are widely used to evaluate RAG systems:

- **Natural Questions (NQ)**, **TriviaQA**, and **WebQuestions** for open-domain QA [52].
- **FEVER** and **AveriTeC** for fact-checking, emphasizing verifiability.
- **KILT** benchmark suite integrates QA, dialog, slot filling, and entity linking over Wikipedia.
- **BEIR** evaluates retrieval across 31 zero-shot tasks in domains like biomedicine and finance.
- **MTRAG** targets multi-turn conversations requiring sequential retrieval and reasoning.
- **TREC RAG Track (2024–)** defines unified evaluation of retrieval, generation, and support quality over MS MARCO with metrics like nugget recall and citation coverage [71].

6.3 Retrieval-Augmented Generation Assessment System

RAGAS (Retrieval-Augmented Generation Assessment System) is an evaluation framework specifically designed for assessing and improving the factuality and grounding of RAG systems. Unlike conventional metrics that measure superficial linguistic overlap, RAGAS emphasizes the alignment between generated content and retrieved documents, providing explicit signals regarding factual correctness and attribution quality. By systematically measuring how well the generated outputs are supported by the retrieved evidence, RAGAS helps identify and penalize hallucinations—instances where the model generates plausible but unsupported statements. Consequently, employing RAGAS during model training or iterative fine-tuning guides RAG systems toward producing outputs firmly grounded in verifiable sources, substantially improving factual accuracy and reducing the incidence of hallucinated information.

Table 1: Evaluation Dimensions, Metrics, Benchmarks, and Tools for RAG Systems

Category	**Metric/Tool**	**Description**
Retrieval Accuracy	Recall@k	Proportion of queries with a relevant doc in top-k results
	MRR	Average inverse rank of the first relevant document
	MAP	Mean precision across all relevant retrieved documents
Generation Quality	Exact Match (EM), F1	Measures overlap with ground-truth answers
	BLEU, ROUGE	N-gram based overlap metrics for generation
	Faithfulness Rate	Consistency with retrieved context (manual/automated)
Efficiency	Latency (retrieval/generation)	Time required for each processing step
	Memory/Compute Usage	Resource demands for model operation
Scalability	Index Size vs. Accuracy	Impact of increasing corpus size on performance
	Adaptability	Ability to incorporate new data without retraining
Benchmarks	Natural Questions, TriviaQA	Open-domain QA evaluation
	FEVER, AveriTeC	Fact-checking datasets
	KILT	Multi-task benchmark over Wikipedia
	BEIR	31-task zero-shot retrieval benchmark
	MTRAG	Multi-turn reasoning benchmark
	TREC RAG Track	Unified eval. of retrieval, generation, citation
Tooling	RAGAS	Evaluates factual consistency and grounding

6.4 Impact of Architecture on Evaluation

Empirical studies demonstrate that architectural components in RAG—chunking, embedding, and re-ranking—directly affect performance across tasks and benchmarks.

Fixed-size chunks, as used in early RAG [52], are prone to semantic fragmentation. Semantic chunking (e.g., *ChunkRAG*) improves retrieval precision and answer accuracy by aligning chunk boundaries with discourse structure [84]. The choice of embedding model (e.g., DPR [45], HyDE [19]) influences Recall@k and downstream generation quality. Similarly, re-ranking strategies (e.g., MonoT5) can boost EM and F1 scores by prioritizing more relevant passages.

In high-performing RAG systems, retrieval fidelity correlates strongly with factual answer quality. Thus, optimizing these architectural stages is essential for achieving competitive results on QA and generation benchmarks.

7 Challenges of RAG

This section discusses the challenges of RAG, cases of manifestation of such challenges in the selected domain of RAG application, and outlines existing solutions and the way forward.

7.1 Technical Challenges in RAG

7.1.1 Retrieval Quality and Relevance

The quality of retrieved documents significantly impacts the accuracy of RAG-generated answers. High recall and precision are critical since poor retrieval leads directly to incorrect or irrelevant answers [61]. Traditional methods like BM25 are limited, often missing relevant texts or returning noisy results [12]. Modern neural retrievers with dense embeddings improve performance but still face issues like vocabulary mismatches, ambiguous queries, and domain-specific terminology. Specialized domain tuning, such as using legal embeddings or medical synonym expansion, can help, but maintenance of these tailored retrievers remains challenging. Determining the optimal number of retrieved passages (k) is also complex. Too few passages limit evidence; too many overwhelm the model and introduce irrelevant context. Approaches like ranking retrieved passages or iterative query reformulation can improve retrieval precision, but add complexity and latency [34, 21].

7.1.2 Latency and Efficiency

RAG inherently increases computational complexity and latency compared to standalone LLMs due to retrieval overhead, vector searches, and expanded context processing. Techniques like approximate nearest neighbor indices (e.g., FAISS, HNSW), caching, model distillation, or lightweight retrievers can reduce latency at the expense of accuracy. Integrating retrieval efficiently with large language models (LLMs) and ensuring rapid responses in real-time scenarios (e.g., customer support) remains a significant challenge [3]. Interestingly, using retrieval can allow smaller models to match the performance of larger models without retrieval (e.g., RETRO, Atlas), reducing model size requirements but shifting complexity to maintaining external knowledge bases and infrastructure.

7.1.3 Integration with Large Language Models

Integrating retrieved evidence effectively with LLMs is subtle. Models may ignore retrieved evidence, especially when internal model knowledge conflicts with external retrieved information, leading to a "tug-of-war" effect [41]. Multiple retrieved documents might create confusion or confirmation bias if they contradict each other. Limited input lengths in transformer-based LLMs exacerbate these integration challenges by forcing truncation or summarization, potentially omitting essential context. Fine-tuning models specifically for retrieval-augmented tasks often yields better integration than simple zero-shot prompting but introduces complexity, especially when using non-differentiable or API-based models that do not support custom training.

7.2 System-Level Challenges

7.2.1 Scalability and Infrastructure

Deploying RAG at scale requires substantial engineering to maintain large knowledge corpora and efficient retrieval indices. Systems must handle millions or billions of documents, demanding significant computational resources, efficient indexing, distributed computing infrastructure, and cost management strategies [21]. Efficient indexing methods, caching, and multi-tier retrieval approaches (such as cascaded retrieval) become essential at scale, especially in large deployments like web search engines.

7.2.2 Freshness and Knowledge Updates

One motivation for RAG is providing current information. However, continuously updating external knowledge bases and retrieval indices is challenging. Domains requiring real-time updates (e.g., finance, healthcare) demand sophisticated data pipelines for incremental updates, possibly frequent re-encoding of documents, and synchronization of retrieval indices. Delays in updates or inconsistencies between the LLM's internal knowledge and newly retrieved data can produce outdated or contradictory answers [61].

7.2.3 Hallucination and Reliability

While RAG reduces LLM hallucinations, it does not eliminate them completely. Models may fabricate or misattribute information if retrieval provides incomplete or partially contradictory context.

Legal domain studies found that RAG significantly reduces hallucinations, but still generates errors at concerning rates [56]. Hallucinations also occur in citation generation, with models occasionally inventing nonexistent references. Strategies such as verifying outputs against retrieved sources or calibrating model confidence are needed, but no approach completely prevents hallucination.

7.2.4 Complex Pipeline and Maintenance

RAG systems comprise multiple components—retrievers, rerankers, indexes, and LLMs—resulting in increased complexity and potential points of failure. Maintenance includes synchronizing knowledge updates, managing access controls, orchestrating prompts, and handling multi-turn dialogues. Robust evaluation methods must assess end-to-end performance, retrieval quality, and faithfulness of model outputs to the evidence [3].

7.3 Ethical and Societal Concerns

7.3.1 Bias and Fairness

RAG inherits biases from both underlying language models and retrieved external data. Biases may manifest through selective use or amplification of biased retrieved evidence, especially in historically biased domains like legal or medical information [7, 99]. Ensuring fairness involves curating inclusive datasets, using diverse retrieval results, and possibly prompting LLMs explicitly toward balanced responses.

7.3.2 Trustworthiness and Misinformation

The quality and reliability of retrieved sources directly impact trustworthiness. If misinformation is retrieved, the model can inadvertently disseminate false information. Even credible sources can become outdated or contextually misapplied, leading users to overly trust synthesized AI responses. Transparency in citing sources, maintaining high-quality data, and incorporating verification methods are crucial safeguards.

7.3.3 Privacy and Security

RAG systems handling sensitive data, especially in enterprise or healthcare contexts, raise serious privacy concerns. Risks include accidental exposure of confidential information and vulnerabilities to prompt injection attacks. Implementing rigorous access control, complying with regulations (e.g., GDPR, HIPAA), transparent data usage policies, and security testing are essential to protect privacy and prevent breaches.

7.3.4 Accountability and Transparency

RAG's use of sourced retrieval provides an advantage for accountability, allowing users to trace AI-generated responses back to evidence. However, inaccurate citations or improper synthesis can mislead users. Ethical deployment involves clearly attributing evidence, providing explanations on request, managing user expectations, and clearly delineating accountability—especially when RAG informs critical decisions. Transparency and accountability require ongoing evaluation, oversight, and mechanisms for user feedback and correction [56].

7.4 Application Domains and Case Studies

To concretely illustrate the discussed challenges, the following section examines Retrieval-Augmented Generation (RAG) applications across three distinct domains: legal, medical, and customer support. Each domain has unique requirements influencing the design and deployment of RAG systems.

7.4.1 Legal Domain

Legal practice inherently involves extensive information retrieval from statutes, regulations, and case precedents, making it particularly suited for RAG applications. Commercial legal AI systems such as those provided by Westlaw and LexisNexis leverage RAG for tasks including legal research and case analysis. Accuracy and reliability are paramount; errors such as citing nonexistent cases can lead to severe professional consequences, underscored by incidents involving fabricated case citations generated by general-purpose LLMs such as ChatGPT. RAG attempts to mitigate this by ensuring assertions trace directly to verified sources.

Key challenges include:

- **Complex Retrieval Requirements**: Legal queries often require precise, multifaceted retrieval considering jurisdiction, topic, and specific legal doctrines. Traditional boolean-based retrieval combined with curated metadata typically addresses these needs, while neural retrieval methods struggle with contextual relevance and precision, occasionally retrieving irrelevant yet semantically similar cases [2].

- **Document Length and Context Limitations**: Legal texts such as court opinions or statutes can be extensive, challenging LLM context windows. Summarizing these documents risks omitting crucial details, thereby potentially misleading practitioners. Techniques to mitigate these risks include retrieving case holdings or headnotes, though these condensed forms may omit critical nuances.

- **Citation Standards**: Lawyers expect citations in standardized formats (e.g., Bluebook in the U.S.). AI-generated citations frequently require extensive formatting refinement and precise pinpoint references to be practically useful.

- **Legal Reasoning and Application**: Legal analysis often involves applying retrieved legal precedents to novel scenarios. Current RAG systems primarily provide raw information, whereas nuanced legal reasoning and judgment remain beyond their current capabilities. RAG tools thus typically function as assistants rather than autonomous legal advisors.

- **Information Freshness and Legal Updates**: Rapidly changing legal frameworks necessitate continuous updating of the retrieval corpus to maintain accuracy. Outdated citations or ignorance of recent rulings can critically undermine legal arguments, underscoring the need for dynamic, regularly updated knowledge sources.

Despite these challenges, RAG promises significant benefits by reducing research time and democratizing access to legal information. Ethically, clear boundaries must exist between supporting lawyers and unauthorized legal practice, necessitating explicit disclosures of limitations and human oversight [56].

7.4.2 Medical Domain

Medical and healthcare applications of RAG involve clinical decision support, patient information summarization, and consumer-facing health advice systems. These applications commonly integrate authoritative medical databases such as PubMed and UpToDate to provide reliable, evidence-based responses.

Domain-specific challenges include:

- **Authoritative Source Management**: Medical RAG systems must exclusively retrieve from rigorously vetted medical literature, clinical guidelines, and patient databases, necessitating careful curation and precise summarization to avoid misinterpretation or oversimplification.

- **Rapid Evolution of Medical Knowledge**: Rapid developments, highlighted during events such as the COVID-19 pandemic, require systems to continuously integrate evolving medical evidence and guidelines. Misrepresentations due to outdated or disproven studies pose significant risks, emphasizing the importance of dynamically updated retrieval sources and structured medical knowledge bases [22].

- **Diagnostic Reasoning Limitations**: Medical diagnostics often require complex reasoning rather than straightforward factual retrieval. Current RAG implementations perform best on factual inquiries (e.g., drug dosages, treatment guidelines), while open-ended diagnostic queries involving symptom analysis remain challenging and risk inappropriate advice.

- **Privacy and Patient Data Security**: Ensuring patient privacy when integrating patient-specific records into RAG systems demands robust data handling protocols to avoid breaches or inappropriate data mixing.

- **Bias in Medical Data**: Historical underrepresentation in clinical trials introduces biases that RAG systems may perpetuate if not addressed proactively through diverse and balanced source curation [99].

- **Regulatory Compliance and Liability**: RAG systems providing medical advice may fall under regulatory oversight such as FDA guidelines, requiring explicit disclaimers and positioning as decision-support tools rather than autonomous advisors.

The medical domain emphasizes the necessity for high factual accuracy, ethical transparency, and rigorous validation processes comparable to clinical trials. Successful deployment hinges on clear delineation between supportive advisory roles and direct patient intervention [64].

7.4.3 Customer Support and Knowledge Bases

RAG systems increasingly automate customer support tasks by retrieving information from internal knowledge bases, FAQs, and troubleshooting guides. While typically lower stakes compared to legal or medical domains, customer support RAG applications encounter distinct practical considerations:

- **Flexible Query Interpretation**: Support queries are often informal or ambiguous. Semantic retrieval effectively handles varied user inputs but requires precise handling to avoid irrelevant answers.
- **Multi-turn Interaction Management**: Effective customer support involves maintaining conversation context and dynamically retrieving information based on prior user interactions. This requires complex dialogue management integrated with retrieval processes.
- **Personalization and Privacy Considerations**: Customer-specific information (e.g., orders, account details) necessitates secure, privacy-aware integration within retrieval processes to avoid data misuse.
- **Knowledge Base Quality and Maintenance**: The utility of RAG systems is directly tied to the quality and currency of support documentation, highlighting the critical need for continual updating and management of the knowledge corpus.
- **User Experience and Tone**: Customer support demands empathetic, conversational responses that match user expectations. Ethical transparency in disclosing the AI nature of interactions is essential.
- **Escalation and Handling Failures**: Systems must effectively identify limits of their capabilities, promptly escalating unresolved issues to human support to maintain customer satisfaction.
- **Metrics for Evaluation**: Success in customer support is evaluated through resolution rates and customer satisfaction metrics, requiring responses that not only provide factual accuracy but practical utility.

Ethical considerations include transparency, data security, job impact, and fairness. RAG implementations also provide valuable feedback for continual improvement of internal support documentation based on real-time user interactions, thereby enhancing both system performance and documentation quality.

7.5 Existing and Potential Solutions

7.5.1 Retrieval Quality

Maintaining high retrieval relevance is critical for effective RAG. Strategies to improve retrieval quality include domain-adaptive training, advanced encoders, and query reformulation methods to address vocabulary mismatches [85]. Employing reranking models further boosts relevance by re-scoring initial retrieval results with deeper contextual analysis, enhancing accuracy at the expense of additional computation [4]. Iterative retrieval and chain-of-thought reasoning represent future directions, breaking down complex queries into simpler sub-queries, thus ensuring relevant information retrieval at each reasoning step [90].

7.5.2 Latency

RAG systems introduce latency due to retrieval processes. Solutions include using efficient nearest-neighbor search structures, such as HNSW graphs, which significantly speed up similarity searches [57]. Caching mechanisms, including multi-level and approximate embedding caches (e.g., RAGCache and Proximity cache), enable reuse of previously retrieved information, drastically reducing retrieval time [40, 8]. Adaptive retrieval methods dynamically balance retrieval complexity based on query difficulty, optimizing overall throughput and reducing latency.

7.5.3 Model Integration

Effective integration between retrieval and generation models remains essential. Methods include joint end-to-end training of retrievers and generators, enhancing mutual compatibility and performance [52]. Architectural integration techniques, such as RETRO's cross-attention mechanism, dynamically incorporate retrieved facts during generation [10]. Alternatively, prompt-based integration treats LLMs as black-boxes, conditioning on retrieved documents without architectural modifications. Future hybrid approaches involving reinforcement learning and selective retrieval aim to optimize when and how

external knowledge is incorporated into generation processes.

7.5.4 Hallucination

Reducing factual hallucinations remains a key focus. RAG inherently mitigates hallucinations by grounding outputs in retrieved evidence [82]. Training models to penalize ungrounded assertions and iterative retrieval within reasoning processes further enhance accuracy [90]. Self-check mechanisms (Self-RAG), where models critique and revise their outputs against retrieval results, significantly reduce hallucinated content [6]. External verification and fact-checking modules complement internal methods, collectively ensuring high factual reliability. For instance, RAG systems to cite sources significantly enhance their reliability by directly linking generated information to supporting evidence. This citation capability plays a crucial role in mitigating the common issue of hallucination, where generative models produce plausible yet inaccurate or fabricated information. By explicitly associating each factual statement with retrieved documents, RAG systems encourage transparency and verifiability, enabling users and downstream processes to quickly assess the accuracy and provenance of claims. Moreover, requiring the model to cite sources during generation inherently promotes grounding outputs in verified data, further reducing the risk of generating unsupported statements [82]. Thus, citation functionality not only enhances user trust but also fosters more disciplined, factually accurate generation, substantially decreasing the likelihood of hallucinated outputs.

7.5.5 Scalability

Scalability challenges arise as knowledge corpora expand. Advanced indexing, distributed retrieval, and approximate nearest neighbor techniques facilitate efficient handling of large-scale knowledge bases [57]. Selective indexing and corpus curation, combined with infrastructure improvements like caching and parallel retrieval, allow RAG systems to scale to massive knowledge repositories. Research indicates that moderate-sized models augmented with large external corpora can outperform significantly larger standalone models, suggesting parameter efficiency advantages [10].

7.5.6 Knowledge Freshness

Rapidly evolving information necessitates regularly updated knowledge bases. RAG systems can efficiently maintain knowledge freshness through incremental updates and selective retrieval methods without requiring frequent retraining [30]. Integrating live search APIs and hybrid retrieval methods ensure real-time information retrieval, addressing dynamic knowledge demands [21]. Continuous updates and user-feedback integration support lifelong learning and timely information access.

7.5.7 Bias

Addressing bias in RAG involves curating balanced knowledge sources, employing diversification techniques in retrieval, and adjusting retriever embeddings to counteract inherent biases [46]. Prompts and model training that encourage balanced representation, along with transparency in source attribution, further mitigate bias propagation. This multi-faceted approach helps minimize biases in RAG outputs.

7.5.8 Misinformation

Combating misinformation involves preventive measures like curating trustworthy knowledge sources and reactive verification through stance classifiers and credibility assessments [66]. Models employing vigilant prompting, cross-verification with multiple retrieved documents, and external fact-checking modules enhance reliability and truthfulness. Robustness against adversarial misinformation insertion through continuous monitoring and data validation further strengthens RAG systems, ensuring accurate information dissemination.

8 Discussion and Future Direction

8.1 Synthesis of Findings

Retrieval-Augmented Generation (RAG) has emerged as a paradigm shift in AI, enabling language models to dynamically retrieve and incorporate external knowledge. Studies confirm that RAG-based models significantly outperform purely parametric generative models, achieving state-of-the-art results on knowledge-intensive NLP tasks [52, 45]. Lewis et al. [52] demonstrated that RAG surpasses fine-tuned BART on open-domain question answering (QA), generating more specific and factual responses.

The historical development of RAG is rooted in the evolution of open-domain QA architectures. Early approaches like DrQA [12] relied on sparse lexical retrieval, whereas modern RAG implementations integrate differentiable dense retrievers such as Dense Passage Retrieval (DPR) [45]. By conditioning generation on retrieved evidence, RAG mitigates issues like hallucinations and stale knowledge [34]. The comparative analysis of different RAG architectures reveals trade-offs in retrieval effectiveness, generative fluency, and computational cost.

Despite continuous improvements, challenges remain. RAG's dependency on retrieval quality makes it vulnerable to retrieval failures. Errors in retrieving relevant documents lead to incorrect outputs. Additionally, integrating multiple retrieved passages introduces challenges in fusion mechanisms, sometimes resulting in contradictory evidence or increased latency. The need for robust retrieval strategies and enhanced fusion methods remains a critical research direction.

8.2 Implications for Proprietary Data

A key development in RAG is its application to enterprise AI, enabling access to proprietary data without embedding sensitive information into model parameters [35]. This architecture supports data privacy and ensures outputs remain grounded in up-to-date knowledge [74]. Unlike traditional LLMs, which require retraining to update internal knowledge, RAG allows retrieval components to be independently refreshed [74].

However, enterprise RAG deployments introduce new security concerns. The retriever could inadvertently expose confidential information if strict access controls are not enforced. Organizations must implement robust authentication mechanisms, ensuring that retrieved documents align with user permissions. Privacy-preserving techniques such as encrypted search indices and federated retrieval [58] are promising solutions to mitigate risks.

The ability of RAG to function over proprietary knowledge bases has made it a preferred choice for industries handling sensitive information, including finance, healthcare, and legal sectors. As enterprises scale RAG systems, optimizing retrieval latency and ensuring regulatory compliance will be paramount.

8.3 Future Research Directions

Key avenues for advancing RAG include:

- **Multi-hop retrieval**: RAG pipelines typically retrieve single passages, but many queries require reasoning over multiple documents. Standard RAG models struggle with multi-hop questions [88], motivating research on multi-step retrieval and reasoning (e.g., chained or iterative RAG methods).
- **Secure and privacy-aware retrieval**: As RAG is applied to sensitive data, privacy becomes critical. Recent work proposes encrypting knowledge bases and embeddings to prevent unauthorized access while preserving performance [107], and integrating differential privacy and secure computation to guard against information leakage [42].
- **Multimodal and agentic RAG**: Extending RAG beyond text can leverage rich context. Multimodal RAG (MRAG) systems incorporate images, video, or other modalities into retrieval and generation, reducing hallucinations and improving performance on visual queries [59]. Separately, *Agentic* RAG embeds autonomous agents into the retrieval pipeline, enabling dynamic planning, tool use, and multi-step reasoning in the loop [83].
- **Structured knowledge integration**: Traditional RAG relies on vector search over unstructured text, which can miss complex relationships. Incorporating structured knowledge (e.g., knowledge graphs) can improve semantic understanding. Surveys and systems demonstrate that linking RAG to knowledge graphs yields more accurate and coherent answers [14, 24].
- **Real-time or streaming retrieval**: Many applications require RAG to operate on live data streams. For example, systems like *StreamingRAG* build evolving knowledge graphs from streaming inputs to provide timely context updates, achieving much faster throughput and higher temporal relevance than static RAG [79].

8.4 Emerging Practical Applications

RAG is finding use in a range of new applications:

- **Enterprise search**: Companies employ RAG to query internal document repositories. By building RAG pipelines around enterprise vector databases, organizations can retrieve and

synthesize answers from private knowledge with strong performance, scalability, and security guarantees [89].

- **Real-time assistants**: Personal or domain-specific AI assistants can use RAG to retrieve current information (e.g., news, stock prices, weather) on the fly. The RAG framework naturally supports grounding LLM responses in real-time data, improving accuracy and user trust [33].

- **Fact-checking**: Integrating RAG into fact-checking pipelines can automate evidence retrieval. Recent systems use RAG-based reasoning (e.g., chain-of-RAG) to fetch and synthesize evidence for claims, improving veracity predictions in political and news domains [1].

- **Conversational agents**: Chatbots and customer-service agents increasingly leverage RAG to provide informed responses. For instance, a RAG-based response system for contact centers has shown higher accuracy and relevance than traditional models [91], helping agents answer queries with up-to-date information.

- **Low-resource QA**: In domains with scarce curated data, RAG can bootstrap question answering by harnessing alternative corpora. A two-layer RAG system using social-media content answered medical questions accurately on modest hardware, demonstrating viability in low-resource settings [17].

9 Conclusion

Retrieval-Augmented Generation (RAG) represents a fundamental advancement in AI, bridging retrieval and generation for improved factuality and adaptability. This review highlights the evolution of RAG from early retrieve-and-read systems to sophisticated architectures integrating neural retrievers and sequence-to-sequence generators. RAG has demonstrated significant benefits across open-domain QA, enterprise applications, and AI-powered search.

While RAG enhances generative AI with dynamic knowledge retrieval, several challenges remain. Ensuring high-quality retrieval, handling conflicting retrieved evidence, and scaling retrieval mechanisms for large knowledge bases require further research. Additionally, security considerations in proprietary deployments necessitate privacy-preserving retrieval strategies.

Future research should focus on optimizing retrieval efficiency, refining document fusion strategies, and developing robust evaluation metrics for retrieval-augmented generation. The continued convergence of information retrieval and AI-powered text generation will define the next generation of intelligent assistants, transforming how users interact with digital knowledge.

Acknowledgments

This work (J.A and A.B) was supported by the University of Tennessee startup funding. The authors acknowledge the use of facilities and instrumentation at the UT Knoxville Institute for Advanced Materials and Manufacturing (IAMM) supported in part by the National Science Foundation Materials Research Science and Engineering Center program through the UT Knoxville Center for Advanced Materials and Manufacturing (DMR-2309083). We extend our gratitude to the faculty and staff of UT-ORII for their invaluable support.

Conflict of Interest

The authors confirm there is no conflict of interest.

References

[1] Mohammed AbdulKhaliq, Paul Yu-Chun Chang, Mingyang Ma, Bernhard Pflugfelder, and Filip Miletic. Ragar, your falsehood radar: Rag-augmented reasoning for political fact-checking using multimodal llms. *arXiv preprint arXiv:2404.12065*, 2024.

[2] Mark Agatonovic, Anastasia Shimorina, et al. Cbr-rag: Case-based reasoning for retrieval augmented generation in llms for legal question answering. In *Proceedings of the 2024 International Conference on Artificial Intelligence and Law (ICAIL)*, 2024 (to appear).

[3] Rama Akkiraju, Anbang Xu, Deepak Bora, Tan Yu, Lu An, Vishal Seth, Aaditya Shukla, et al. FACTS about building retrieval augmented generation-based chatbots. arXiv preprint arXiv:2407.07858, 2024.

[4] Yuwei An, Yihua Cheng, Seo Jin Park, and Junchen Jiang. Hyperrag: Enhancing quality-efficiency tradeoffs in retrieval-augmented generation with reranker kv-cache reuse. arXiv preprint arXiv:2504.02921, 2025.

[5] 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*, 2023.

[6] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[7] Ricardo Baeza-Yates. Bias on the web. *Communications of the ACM*, 61(6):54–61, 2018.

[8] Shai Bergman, Zhang Ji, Anne-Marie Kermarrec, Diana Petrescu, Rafael Pires, Mathis Randl, and Martijn de Vos. Leveraging approximate caching for faster retrieval-augmented generation. In *Proceedings of the 5th Workshop on Machine Learning and Systems (EuroMLSys)*, 2025.

[9] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2022.

[10] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*. PMLR, 2022.

[11] Isabelle Bousquette. Ai doesn't know much about golf. or farming. or mortgages. or ... *The Wall Street Journal*, 2024.

[12] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1870–1879, 2017.

[13] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762, 2024.

[14] Ruixi Chen. Retrieval-augmented generation with knowledge graphs: A survey. *arXiv preprint arXiv:2511.00000*, 2025. CSUC 2025 submission.

[15] Xiuyi Chen, Fandong Meng, Peng Li, Feilong Chen, Shuang Xu, Bo Xu, and Jie Zhou. Bridging the gap between prior and posterior knowledge selection for knowledge-grounded dialogue generation. In *EMNLP*, 2020.

[16] Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. Lift yourself up: Retrieval-augmented text generation with self-memory. *Advances in Neural Information Processing Systems*, 36:43780–43799, 2023.

[17] Sudeshna Das, Yao Ge, Yuting Guo, Swati Rajwal, JaMor Hairston, Jeanne Powell, Drew Walker, Snigdha Peddireddy, Sahithi Lakamana, Selen Bozkurt, Matthew Reyna, Reza Sameni, Yunyu Xiao, Sangmi Kim, Rasheeta Chandler, Natalie Hernandez, Danielle Mowery, Rachel Wightman, Jennifer Love, Anthony Spadaro, Jeanmarie Perrone, and Abeed Sarker. Two-layer retrieval-augmented generation framework for low-resource medical question answering using reddit data: Proof-of-concept study. *J. Med. Internet Res.*, 27:e66220, 2025.

[18] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[19] Yujuan Ding, Wenqi Fan, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meets llms: Towards retrieval-augmented large language models. *arXiv preprint arXiv:2405.06211*, 2024.

[20] Matouš Eibich, Shivay Nagpal, and Alexander Fred-Ojala. ARAGOG: Advanced RAG output grading. *arXiv preprint arXiv:2404.01037*, 2024.

[21] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997, 2024.

[22] Stephen Gilbert, Jakob N. Kather, and Aidan Hogan. Augmented non-hallucinating large language models as medical information curators. *NPJ Digital Medicine*, 7(1):100030, 2024.

[23] Aditya Golatkar, Alessandro Achille, Luca Zancato, Yu-Xiang Wang, Ashwin Swaminathan, and Stefano Soatto. Cpr: Retrieval augmented generation for copyright protection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12374–12384, 2024.

[24] Xiaoming Guo, Shengting Cao, Shenglin Li, Qi Yin, and Cien Li. Structugraphrag: Structured document-informed knowledge graphs for retrieval-augmented generation. In *AAAI Spring Symposium on Generative AI for Social Science Research*, 2024.

[25] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 3929–3938, 2020.

[26] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pages 3929–3938, Online, 2020. PMLR.

[27] Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints*, 3, 2023.

[28] Haoyu Han, Harry Shomer, Yu Wang, Yongjia Lei, Kai Guo, Zhigang Hua, Bo Long, Hui Liu, and Jiliang Tang. Rag vs. graphrag: A systematic evaluation and key insights. *arXiv preprint arXiv:2502.11371*, 2025.

[29] Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A. Rossi, Subhabrata Mukherjee, Xianfeng Tang, Bo Long, Tong Zhao, Neil Shah, Yinglong Xia, and Jiliang Tang. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309*, 2025.

[30] Guoxiu He, Xin Song, and Aixin Sun. Knowledge updating? no more model editing! just selective contextual reasoning. *Journal of the ACM*, 2025. to appear.

[31] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907, 2024.

[32] IBM. What is rag (retrieval augmented generation)? *IBM*, 2023.

[33] IBM Research. What is retrieval-augmented generation? IBM Research Blog, 22 August 2023, 2023. https://research.ibm.com/blog/retrieval-augmented-generation-RAG.

[34] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 874–880, 2021.

[35] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

[36] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24:1–43, 2023.

[37] Gautier Izacard, Xin Wan, Christoph Böhm, Kazuma Irie, Nathanael Schärli, and Sebastian Riedel. Unsupervised dense information retrieval with contrastive learning. *Transactions of the Association for Computational Linguistics*, 2022. arXiv:2201.10672.

[38] Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore, December 2023. Association for Computational Linguistics.

[39] Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O Arik. Long-context llms meet rag: Overcoming challenges for long inputs in rag. In *The Thirteenth International Conference on Learning Representations*, 2024.

[40] Chao Jin, Zili Zhang, Xuanlin Jiang, Fangyue Liu, Xin Liu, Xuanzhe Liu, and Xin Jin. Ragcache: Efficient knowledge caching for retrieval-augmented generation. arXiv preprint arXiv:2404.12457, 2024.

[41] Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, Xiaojian Jiang, Jiexin Xu, Qiuxia Li, and Jun Zhao. Tug-of-war between knowledge: Exploring and resolving knowledge conflicts in retrieval-augmented language models. In *Proceedings of the 2024 Joint Conference on Computational Language Learning and Language Resources and Evaluation (LREC-COLING)*, 2024.

[42] Sheshananda Reddy Kandula. Securing retrieval-augmented generation: Privacy risks and mitigation strategies. *SSRN Electronic Journal*, 2025.

[43] Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation (surge). *arXiv preprint arXiv:2305.18846*, 2023.

[44] Vladimir Karpukhin, Barlas Oguz, 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 (EMNLP)*, pages 6769–6781, 2020.

[45] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 (EMNLP)*, pages 6769–6781, Online, 2020. Association for Computational Linguistics.

[46] Taeyoun Kim, Jacob Springer, Aditi Raghunathan, and Maarten Sap. Mitigating bias in rag: Controlling the embedder. arXiv preprint arXiv:2502.17390, 2025.

[47] Lilly Kumari, Usama Bin Shafqat, and Nikhil Sarda. Retrieval-augmented generation for dialog modeling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS 2023) Workshops*, 2023.

[48] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6086–6096, Florence, Italy, 2019. Association for Computational Linguistics.

[49] Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. Pre-training via paraphrasing. *arXiv:2006.15020*, 2020.

[50] Patrick Lewis. Time100 ai 2024. *Time*, 2024.

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

[52] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[53] Yangning Li, Yinghui Li, Xinyu Wang, Yong Jiang, Zhen Zhang, Xinran Zheng, Hui Wang, Hai-Tao Zheng, Pengjun Xie, Philip S. Yu, and Jingren Zhou. Benchmarking multimodal retrieval augmented generation with dynamic VQA dataset and self-adaptive planning agent. *arXiv preprint arXiv:2411.02937*, 2024.

[54] Zhen Li, Cheng Li, Ming Zhang, Qiaozhu Mei, and Michael Bendersky. Retrieval augmented generation or long-context LLMs? a comprehensive study and hybrid approach. *arXiv preprint arXiv:2407.16833*, 2024.

[55] Xun Liang, Simin Niu, Zhiyu Li, Sensen Zhang, Hanyu Wang, Feiyu Xiong, Jason Zhaoxin Fan, Bo Tang, Shichao Song, Mengwei Wang, and Jiawei Yang. Saferag: Benchmarking security in retrieval-augmented generation of large language models. *arXiv preprint arXiv:2501.18636*, 2025.

[56] Varun Magesh, Faiz Surani, Matthew Dahl, Mirac Suzgun, Christopher D. Manning, and Daniel E. Ho. Hallucination-free? assessing the reliability of leading ai legal research tools. *Journal of Empirical Legal Studies*, 0(1):1–27, 2025. Early View, https://doi.org/10.1111/jels.12413.

[57] Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9):2225–2237, 2018.

[58] Priyanka Mary Mammen. Federated learning: Opportunities and challenges. *arXiv preprint arXiv:2101.05428*, 2021. Accessed: 2025-03-19.

[59] Lang Mei, Siyu Mo, Zhihan Yang, and Chong Chen. A survey of multimodal retrieval-augmented generation. *arXiv preprint arXiv:2504.08748*, 2025.

[60] Rick Merritt. What is retrieval-augmented generation aka RAG? NVIDIA Blog, 15 November 2023, 2023. https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/.

[61] Jing Miao, Charat Thongprayoon, Supawadee Suppadungsuk, Oscar A. Garcia Valencia, and Wisit Cheungpasitporn. Integrating retrieval-augmented generation with large language models in nephrology: Advancing practical applications. *Medicina (Kaunas)*, 60(3):445, 2024.

[62] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2022.

[63] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) (Demonstrations)*, pages 72–77, 2019.

[64] Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*, 2023.

[65] NVIDIA. Activate your data with custom generative ai. *NVIDIA*, 2023.

[66] Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. On the risk of misinformation pollution with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1389–1403, 2023.

[67] Md Rizwan Parvez, Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. Retrieval augmented code generation and summarization. In *Findings of EMNLP 2021*, pages 2719–2734, 2021.

[68] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online, June 2021. Association for Computational Linguistics.

[69] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. Kilt: a benchmark for knowledge intensive language tasks. In *Proceedings of NAACL-HLT 2021*, pages 2523–2544, 2021.

[70] Ronak Pradeep, Nandan Thakur, Sahel Sharifymoghaddam, Eric Zhang, Ryan Nguyen, Daniel Campos, Nick Craswell, and Jimmy Lin. Ragnarök: A reusable RAG framework and baselines for the TREC 2024 retrieval-augmented generation track. *arXiv preprint arXiv:2406.16828*, 2024.

[71] Ronak Pradeep, Nandan Thakur, Sahel Sharifymoghaddam, Eric Zhang, Ryan Nguyen, Daniel Campos, Nick Craswell, and Jimmy Lin. Ragnarök: A reusable retrieval-augmented generation framework and baselines for the TREC 2024 RAG track. *arXiv preprint arXiv:2406.16828*, 2024. Introduces the AutoNuggetizer evaluation and shows its high correlation with human judgments.

[72] Irina Radeva, Ivan Popchev, Lyubka Doukovska, and Miroslava Dimitrova. Web application for retrieval-augmented generation: Implementation and testing. *Electronics*, 13(7):1361, 2024.

[73] 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. Introduces the T5 model family, including T5-Large used as a closed-book QA baseline.

[74] AWS AI Research. Enterprise ai with retrieval-augmented generation. *AWS Blog*, 2023.

[75] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *EMNLP*, 2020.

[76] Devendra Singh Sachan, Siva Reddy, William L. Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. *CoRR*, abs/2106.05346, 2021.

[77] Devendra Singh Sachan, Siva Reddy, William L. Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL 2021)*, pages 2074–2088, Online, 2021. Association for Computational Linguistics.

[78] Alireza Salemi and Hamed Zamani. Evaluating retrieval quality in retrieval-augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2395–2400, 2024.

[79] Murugan Sankaradas, Ravi K. Rajendran, and Srimat Chakradhar. Streamingrag: Real-time contextual retrieval and generation framework. *arXiv preprint arXiv:2501.14101*, 2025.

[80] Kurt Shuster, Da Ju, Peng Xu, Eric Michael Smith, Emily Dinan, Stephen Roller, Piali Koura, Y-Lan Boureau, and Jason Weston. Blenderbot 3: A deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.

[81] Kurt Shuster, Mojtaba Komeili, Leonard Adolphs, Stephen Roller, Arthur Szlam, and Jason Weston. Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 373–393. Association for Computational Linguistics, 2022.

[82] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, 2021.

[83] Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. Agentic retrieval-augmented generation: A survey on agentic rag. *arXiv preprint arXiv:2501.09136*, 2025.

[84] Ishneet S. Singh, Ritvik Aggarwal, Ibrahim Allahverdiyev, Muhammad Taha, Aslihan Akalin, Kevin Zhu, and Sean O'Brien. Chunkrag: A novel llm-chunk filtering method for rag systems. *arXiv preprint arXiv:2410.19572*, 2025.

[85] Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. Improving the domain adaptation of retrieval augmented generation (rag) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17, 2023.

[86] Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 12–22, 2024.

[87] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, pages 2440–2448, 2015.

[88] Yixuan Tang and Yi Yang. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. In *COLM*, 2024. OpenReview preprint.

[89] Harvey Team. Enterprise-grade rag systems: High-performance rag with vector databases, 2025. Harvey AI blog.

[90] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 10014–10037, 2023.

[91] Sriram Veturi, Saurabh Vaichal, Reshma Lal Jagadheesh, Nafis Irtiza Tripto, and Nian Yan. Rag based question-answering for contextual response prediction system. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024.

[92] Shuohang Wang, Mo Yu, Xinya Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, and Jing Jiang. R^3: Reinforced reader-ranker for open-domain question answering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 5981–5988, 2018.

[93] Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, et al. Searching for best practices in retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17716–17736, 2024.

[94] Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. Learning to filter context for retrieval-augmented generation. *arXiv preprint arXiv:2311.08377*, 2023.

[95] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *3rd International*

Conference on Learning Representations (ICLR), Conference Track Proceedings, 2015.

[96] Di Wu, Jia-Chen Gu, Kai-Wei Chang, and Nanyun Peng. Self-routing rag: Binding selective retrieval with knowledge verbalization. *arXiv preprint arXiv:2504.01018*, 2025.

[97] Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. Benchmarking retrieval-augmented generation for medicine. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 6233–6251, 2024.

[98] Diji Yang, Linda Zeng, Jinmeng Rao, and Yi Zhang. Knowing you don't know: Learning when to continue search in multi-round rag through self-practicing. In *Proceedings of SIGIR 2025*, 2025. arXiv:2505.02811.

[99] Ruiyang Yang, Xin Huang, Xingyu Li, et al. Retrieval-augmented generation for generative artificial intelligence in health care. *npj Health Systems*, 2(2):1–8, 2025.

[100] Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *Advances in Neural Information Processing Systems*, 37:121156–121184, 2024.

[101] Shenglai Zeng, Jiankun Zhang, Pengfei He, Yiding Liu, Yue Xing, Han Xu, Jie Ren, Yi Chang, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4505–4524. Association for Computational Linguistics, 2024.

[102] Wenjia Zhai. Self-adaptive multimodal retrieval-augmented generation. *arXiv preprint arXiv:2410.11321*, 2024.

[103] Ruichen Zhang, Hongyang Du, Yinqiu Liu, Dusit Niyato, Jiawen Kang, Sumei Sun, Xuemin Shen, and H Vincent Poor. Interactive ai with retrieval-augmented generation for next generation networking. *IEEE Network*, 2024.

[104] 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. In *First Conference on Language Modeling*, 2024.

[105] Danyang Zhao. FRAG: Toward federated vector database management for collaborative and secure retrieval-augmented generation. arXiv preprint arXiv:2410.13272, 2024. https://doi.org/10.48550/arXiv.2410.13272.

[106] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*, 2024.

[107] Pengcheng Zhou, Yinglun Feng, and Zhongliang Yang. Privacy-aware rag: Secure and isolated knowledge retrieval. *arXiv preprint arXiv:2503.15548*, 2025.