Enhancing Regulatory Compliance QA via Hierarchical Semantic Chunking and Domain-Adaptive Reranking

Anonymous EMNLP submission

Abstract

As AI-enabled research accelerates pharmaceutical technological advances, legislation and regulation worldwide are evolving rapidly and often have a significant impact. Compliance with fragmented, frequently updated national regulations presents a pressing challenge for multinational organizations, a global leader in pharmaceuticals and diagnostics. This project proposes an AI-powered interactive dialogue system, which streamlines the interpretation and alignment of the evolving regulatory requirements that directly impact a pharmaceutical company's internal standards. We introduce HiSACC, a hierarchical semantic chunking method, and BGE-Reranker, a domain-adaptive semantic re-ranking model using fine-tuning, which are designed to optimize the chunking and re-ranking processes, ensuring more accurate and context-aware responses to regulatory queries.

The system leverages advanced Large Language Models (LLMs) to generate user-specific responses and incorporates Retrieval-Augmented Generation (RAG) technology to enable precise, context-aware responses to complex medical-legal queries, minimizing the occurrence of hallucination biases. This project is an innovative tool designed to reduce manual workload and improve the efficiency and precision of navigating regulatory compliance. Beyond pharmaceuticals, the system's adaptable framework holds promise for other domains that experience frequent legal updates, particularly in banking, finance, data privacy, and cybersecurity. By adapting the system to address regulatory challenges in these sectors, organizations can ensure adherence to the latest legal standards, thereby mitigating risks and enhancing operational effectiveness across various industries[1].

[1]Anonymous implementation and evaluation scripts are available at https://anonymous.4open.science/r/hisacc-bge-ABCC/.

1 Introduction

The pharmaceutical industry operates within a highly regulated environment due to its products directly impacting human health. Ensuring legal compliance is crucial to safeguard patient safety, maintain public trust, and provide safe and effective medications (Kher, 2020). Non-compliance can lead to significant financial losses, legal penalties, and reputation damage. In 2023, the U.S. Food and Drug Administration (FDA) issued 1,150 warning letters regarding drug compliance issues (Sharma et al., 2023), and in 2024, the average cost per violation reached $14.8 million. For pharmaceutical companies, maintaining compliance that exceeds industry standards is not just about avoiding fines; but it is essential for their survival in a highly competitive market.

Regulations continue to evolve rapidly to address advancements in biotechnology and market trends. In 2024, the FDA revised 15% of drug manufacturing regulations to better adapt to new biotechnological advances (U.S. Food and Drug Administration, 2024). However, this rapid regulatory change has led to an increasing shortage of skilled professionals capable of managing the complexity of regulatory requirements. A recent survey showed that 56% of pharmaceutical companies reported difficulties in hiring such talent (ComplianceQuest, 2025). This situation has led to an increased reliance on automation tools to handle intricate compliance processes, such as Document Management Systems (DMS) and compliance software, which help automate tasks like record-keeping, risk assessment, and monitoring of regulatory changes (Jordan et al., 2022). Yet, despite the availability of these tools, experts and compliance officers still struggle to efficiently track and adapt to continuous updates across global jurisdictions and diverse industry segments.

To address these challenges, this study proposes

the development of an AI-driven interactive dialogue system powered by Large Language Models (LLMs). These models are capable of processing vast amounts of legal documents and regulatory updates, helping businesses understand and respond to changes in a shorter time frame. Specifically, the system parses regulatory texts, guidance documents, and industry standards, assisting companies in adapting to evolving regulatory requirements across different global regions. However, one critical issue limiting LLM is the tendency to hallucination biases, where the model generates plausible but incorrect information (Ji et al., 2023). In high-stakes industries like pharmaceuticals, even minor inaccuracies in regulatory interpretations can result in noncompliance, legal penalties, and irreversible harm to patient safety.

To mitigate the hallucination bias of LLM, we introduce a solution based on Retrieval-Augmented Generation (RAG) technology. In collaboration with an industrial partner, we leverage their internal compliance reports, Standard Operating Procedures (SOP) documents, and quality control records to build a system that integrates two key innovations: HiSACC, a hierarchical semantic chunking method, and BGE-Reranker, a domain-adaptive semantic re-ranking model using fine-tuning. HiSACC optimizes the chunking process by dynamically identifying semantically meaningful segments in regulatory documents, while BGE-Reranker enhances retrieval performance by improving post-retrieval ranking to better match query contexts. By optimizing both pre-retrieval filtering and post-retrieval validation algorithms, our system significantly improves the accuracy and timeliness of compliance checks. This not only improves operational efficiency but also reduces the risk of regulatory penalties and enhances the reliability of compliance-related decisions.

The effectiveness of our system will be rigorously evaluated by compliance officers to ensure the alignment with the industry's zero-tolerance standards for regulatory inaccuracies. Beyond pharmaceuticals, the system's adaptable framework holds promise for other domains that face similar regulatory challenges, such as banking, finance, data privacy, and cybersecurity, where dynamic regulatory landscapes demand high precision and real-time adaptability.

2 Related Work

2.1 Hallucination Mitigation in Language Models

In regulatory compliance applications, ensuring the accuracy of generated information is paramount. Despite significant advancements in large language models (LLMs), these models are prone to hallucination bias, where they produce inaccurate or inconsistent content (Ji et al., 2022). To address this issue, various strategies have been proposed, including Prompt Engineering, Retrieval-Augmented Generation (RAG), and Fine-Tuning. Prompt Engineering involves crafting specific input prompts to guide the model, but it lacks generalizability across tasks (Brown et al., 2020). Fine-Tuning adapts models to specific domains through training on specialized data, improving performance, but it is costly and not adaptable to rapid regulatory changes (Wei et al., 2022). In contrast, RAG integrates real-time information retrieval during text generation, reducing hallucination bias and allowing dynamic adaptation to regulatory updates (Lewis et al., 2020). This makes RAG particularly effective for scenarios requiring real-time updates and high accuracy in regulatory compliance. However, naive RAG struggles with complex queries (Gao et al., 2024) and has room for improvement in handling diverse document types and formats.

2.2 Enhancements to RAG Systems

To improve naive RAG, researchers have focused on refining various stages of the process, including Chunking, Pre-Retrieval, and Post-Retrieval.

2.2.1 Chunking Strategies

Traditional chunking methods are rule-based and focus on segmenting text according to predefined boundaries like line breaks, punctuation, or fixed-length paragraphs. While these methods prioritize sentence boundaries and ensure token count limits, they can overlook semantic coherence, which may lead to the truncation of important information or the combination of unrelated text fragments (Gao et al., 2024). As a result, more advanced chunking methods have emerged, such as dynamic chunking strategies. One such approach, LumberChunker, dynamically adjusts chunk boundaries by analyzing semantic shifts in text (Duarte et al., 2024). By employing a pre-trained LLM, LumberChunker identifies points where semantic changes are significant, ensuring that each chunk is semantically co-

herent and independent. Similarly, Meta-Chunking enhances chunking by analyzing text logic and structure using strategies like Margin Sampling Chunking and Perplexity Chunking (Zhao et al., 2024). These methods improve semantic segmentation by considering perplexity levels and structural logic, which helps identify appropriate chunk boundaries. For documents with rich visual content, VisRAG leverages a visual language model (VLM) to process text alongside layout and image data, determining chunk boundaries based on visual elements (Yu et al., 2024). These dynamic and multimodal chunking strategies significantly enhance the retrieval process by ensuring that the retrieved text segments are both relevant and semantically complete.

2.2.2 Pre-Retrieval Optimization

Pre-retrieval optimization plays a critical role in refining the query and index structures to maximize retrieval accuracy. Techniques in this stage include query rewriting, expansion, and transformation, which align user queries with the indexed data more effectively, ensuring more relevant retrieval results.

Query Rewriting Query rewriting focuses on adjusting rare or specific queries to improve search engine performance, particularly for long-tail queries (Wang et al., 2024b). This approach is crucial for ensuring that specialized user intents are captured more effectively during information retrieval.

Query Expansion Query expansion involves adding synonyms, hypernyms, or related terms to a query, broadening its scope and improving retrieval relevance (Koo et al., 2024). This strategy is especially beneficial in open-domain question-answering tasks, where expanding the query helps retrieve a more comprehensive set of relevant documents.

Query Transformation Query transformation alters the structure or semantics of a query while maintaining its original intent, enhancing its alignment with the requirements of LLMs (Chan et al., 2024). This modification may include adding qualifiers or reordering terms to optimize the query's effectiveness in the retrieval process.

2.2.3 Post-Retrieval Optimization

Once relevant documents are retrieved, post-retrieval optimization is essential to refine the retrieved data and ensure it enhances the quality of the generated response. This process includes strategies for efficiently managing the retrieved fragments and optimizing their relevance for final generation.

Efficient Compression In naive RAG, the Approximate Nearest Neighbor (ANN) algorithm retrieves document blocks most similar to the query and ranks them by similarity (Wang et al., 2024a). To avoid information overload and improve generation accuracy, systems like COCOM use a Transformer-based architecture to compress multiple text blocks into a dense embedding vector. This compressed context is then used to generate the final response (Rau et al., 2024). By merging multiple fragments into one context vector, the system reduces sensitivity to irrelevant information and ensures more focused generation.

Precise Filtration The post-retrieval filtration process involves further refining the retrieved text blocks to ensure only the most relevant fragments are used for generation. Models like E2E-AFG utilize an end-to-end adaptive filtering mechanism to identify and filter out irrelevant text (Jiang et al., 2024). Techniques such as String Inclusion, Lexical Overlap, and Conditional Cross-Mutual Information (CXMI) help determine the relevance of text blocks by analyzing their alignment with the generated pseudo-answer (Es et al., 2023). Furthermore, models like Shi et al. incorporate Abstract Meaning Representation (AMR) to filter out noise and focus the model's attention on key concepts, improving accuracy, especially in specialized domains (Shi et al., 2024).

3 Problem Definition

The core objective of this research is formalized as a Regulatory-Compliance Question Answering (RC-QA) task within retrieval-augmented generation (RAG) frameworks. It is specifically aimed at optimizing the retrieval and generation processes for regulatory compliance dialogue systems in high-stakes pharmaceutical environments. Given the rapidly evolving regulatory landscape, multinational pharmaceutical enterprises face significant challenges in ensuring compliance with complex, diverse, fragmented, and frequently updated regulations across multiple jurisdictions. These challenges arise due to the substantial volume of information, diversity in document formats, and the intricacies involved in maintaining accuracy and

timeliness. These conditions necessitate solutions that meet the practical and stringent demands of real-world industrial settings.

3.1 Task Definition

Input: The input consists of a corpus $D = \{d_1, d_2, \ldots, d_n\}$ containing diverse regulatory documents, guidelines, compliance records, and a continuously evolving set of various internal documentation, such as Standard Operating Procedures (SOPs), audit reports, and regulatory guidance documents. Each document d_i within the corpus may include structured, semi-structured, or unstructured data, undergoing secure ingestion and metadata normalization processes.

Query: A natural language query q posed by a compliance officer seeking precise regulatory information, operational guidance, or compliance interpretations.

System Objective: Given a query q, the system must retrieve context segments $C = \{c_1, c_2, \ldots, c_k\}$ from the corpus D, with relevance assessed by semantic similarity metrics, notably Context Relevance (CR), and validated through accurate File ID Match (FIM). The retrieval process aims to identify minimal yet highly relevant evidence passages, ensuring high CR, precise identification of source documents (FIM), extensive coverage of authoritative regulatory content (Context Coverage (CC)), and minimal inclusion of irrelevant information to reduce the Over-Retrieval Penalty (ORP).

Based on these retrieved context segments, the system is tasked with generating coherent, accurate, and contextually grounded natural language responses r that precisely answer the query q. Response quality is quantitatively assessed using metrics such as Answer Relevance (AR), Answer Source Match (ASM), and Language Fluency (LF). All generated tokens must have verifiable grounding in the retrieved context, with ungrounded tokens considered hallucinations and penalized accordingly. These constraints are quantified by the Groundedness Rate (GR) and Faithfulness Test (FT).

To operationalize this objective, we formulate the RC-QA task as a multi-metric optimization problem evaluated by a comprehensive set of automatic metrics:

$$F = \{\text{AR}, \text{CR}, \text{GR}, \text{FIM}, \text{CC}, \text{ASM}, \text{LF}, \text{FT}, \text{ORP}\}$$

The overall optimization goal is defined as:

$$\max_{\Theta} \ \mathbb{E}_{(q,r^*,E^*)\sim D_{\text{eval}}} \ w^\top F(r_\Theta(q), E_\Theta(q))$$

where w represents a weight vector prioritizing compliance-critical metrics such as AR, GR, and FIM, and Θ denotes the tunable parameters governing retrieval and response generation.

4 Methodology

4.1 Baseline System Architecture

To facilitate the development and evaluation of improved retrieval and generation methods, we present a complete end-to-end baseline architecture grounded in the standard Retrieval-Augmented Generation (RAG) paradigm (Figure 1). Our system integrates secure document ingestion, semantic indexing, and context-aware response generation into a unified pipeline.

The process begins with secure synchronization of regulatory documents from internal repositories, supported by an incremental update mechanism that processes only new, modified, or deleted files. Documents are then parsed and normalized to produce a consistent textual representation across heterogeneous sources. The semantic preprocessing module segments text into chunks using a recursive strategy based on token limits and structural cues. Each chunk is embedded into a dense vector space and indexed for similarity-based retrieval. During inference, user queries are embedded and matched to relevant chunks, which are concatenated with the query and passed to a language model for grounded response generation.

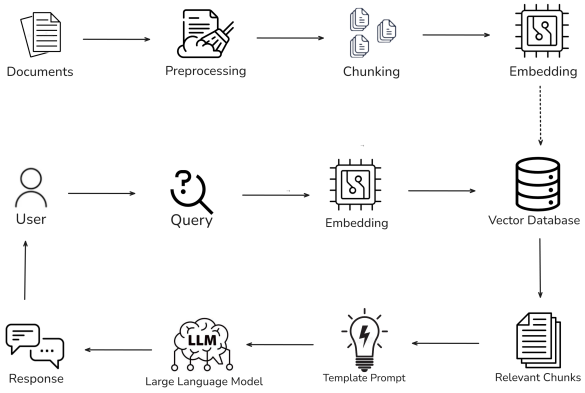

Figure 1: Overview of the baseline Retrieval-Augmented Generation (RAG) system architecture.

4.2 Chunking Optimization with HiSACC

To address semantic fragmentation from traditional splitting methods, we propose HiSACC (Hierarchical Semantic Aggregation for Contextual

Chunking). HiSACC optimizes semantic coherence through hierarchical semantic aggregation.

Initially, minimal semantic units $S = \{s_1, s_2, \ldots, s_k\}$ are embedded into vectors $V = \{v_1, v_2, \ldots, v_k\}$ using a semantic encoder (e.g., Sentence-BERT). Semantic similarity between adjacent vectors is calculated by:

$$M_{i,i+1} = \frac{v_i \cdot v_{i+1}}{\|v_i\|\|v_{i+1}\|}$$

Adjacent segments with similarity $M_{i,i+1} \geq \theta$ are aggregated into initial local semantic groups.

In the hierarchical merging stage, these initial segments $G = \{G_1, G_2, \ldots, G_p\}$ are further merged based on a global semantic coherence threshold γ. A skip-window of size w evaluates average inter-group semantic similarity:

$$\frac{1}{|G_a||G_b|} \sum_{v_i \in G_a} \sum_{v_j \in G_b} \frac{v_i \cdot v_j}{\|v_i\|\|v_j\|} \geq \gamma$$

Segments meeting this criterion merge to produce semantically cohesive chunks.

4.3 Post-Retrieval Optimization via BGE-Reranker

To improve the semantic ranking of retrieved passages, we adopt **BGE-Reranker**[2] (Xiao et al., 2023), a cross-encoder model that jointly encodes query-document pairs for fine-grained relevance estimation.Given a query q and candidates $\{d_1, \ldots, d_k\}$, it computes matching scores:

$$s_i = f_\theta(q, d_i)$$

To capture global ranking structure, we fine-tune the model using a **listwise loss** (Cao et al., 2007) based on softmax-normalized scores:

$$P(d_i \mid q) = \frac{e^{s_i}}{\sum_{j=1}^{k} e^{s_j}}, \quad \mathcal{L} = -\sum_{i=1}^{k} y_i \log P(d_i \mid q)$$

Here, y_i denotes the relevance label. This approach encourages globally consistent rankings and is well-suited to domains with subtle semantic distinctions.

4.4 Evaluation Framework

Our evaluation rigorously quantifies retrieval and generation quality using cosine similarity over fixed embeddings (Es et al., 2025). Specifically, we define *Answer Relevance (AR)* as $\mathrm{Sim}(q, r)$, *Context Relevance (CR)* as $\mathrm{Sim}(q, c)$, and *Groundedness Rate (GR)* as $\mathrm{Sim}(r, c)$, where

$$\mathrm{Sim}(x, y) = \frac{\phi(x) \cdot \phi(y)}{\|\phi(x)\|\|\phi(y)\|}.$$

File ID Match (FIM) FIM is a binary metric verifying whether the source file ID id^* is among the retrieved file IDs R:

$$\mathrm{FIM}(R, id^*) = \mathbb{I}[id^* \in R].$$

Context Coverage (CC) CC evaluates the maximum semantic similarity between the retrieved contexts $\{c_i\}$ and the authoritative source text s:

$$\mathrm{CC}(\{c_i\}, s) = \max_i \left(\frac{\phi(c_i) \cdot \phi(s)}{\|\phi(c_i)\|\|\phi(s)\|} \right).$$

Answer Source Match (ASM) ASM measures semantic alignment between the generated response r and the reference answer s:

$$\mathrm{ASM}(r, s) = \frac{\phi(r) \cdot \phi(s)}{\|\phi(r)\|\|\phi(s)\|}.$$

Language Fluency (LF) LF is scored using a fluency function (Kim and Kim, 2024) $\psi(\cdot)$ normalized to $[0, 1]$:

$$\mathrm{LF}(r) = \frac{\psi(r)}{10}.$$

Faithfulness Test (FT) FT evaluates how many factual statements in the answer $S = \{s_j\}$ are supported by retrieved contexts (Maynez et al., 2020) $C = \{c_i\}$:

$$\mathrm{FT}(r, C) = \frac{1}{|S|} \sum_{s_j \in S} \mathbb{I}\left[\exists c_i \in C : s_j \in c_i\right].$$

Over-Retrieval Penalty (ORP) ORP penalizes the proportion of retrieved contexts not semantically similar to the source answer. Let τ be the similarity threshold:

$$\mathrm{ORP}(\{c_i\}, s) = 1 - \frac{|\{c_i : \mathrm{CC}(c_i, s) > \tau\}|}{|\{c_i\}|}.$$

5 Experiment and Results

5.1 System Implementation

Data Acquisition and Synchronization. Regulatory documents are sourced from enterprise-level Google Shared Drives using the Google Drive

[2]We use the publicly available `bge-reranker-base` model from Hugging Face: https://huggingface.co/BAAI/bge-reranker-base

API with OAuth 2.0 authentication for secure access (Google Developers, 2024b,a)[3]. To enable efficient incremental updates, a local SQLite database tracks document metadata. During each synchronization cycle, the system compares metadata to detect newly added, updated, or removed files, minimizing redundant processing.

Parsing and Normalization. The system supports multi-format documents, including PDF, DOC/DOCX, XLS/XLSX, CSV, and TXT. Text is extracted using format-specific tools, and all content is standardized into structured text blocks for downstream compatibility.

Semantic Segmentation. A hierarchical recursive chunking strategy is employed to divide documents into semantically coherent units. Paragraph delimiters and whitespace are prioritized, and overlapping windows are used to retain contextual continuity within a preset token limit.

Embedding and Indexing. Text segments are embedded into a high-dimensional semantic space using an internal embedding model hosted on an internal Galileo AI Platform, based on OpenAI's embedding family and served via the Azure OpenAI Service. Embeddings and metadata are stored in a Milvus vector database.

Retrieval and Generation Pipeline. Users interact through a Gradio-based web interface[4], served by a FastAPI backend with Uvicorn for high-concurrency support. Upon query submission, the system retrieves relevant chunks from Milvus, which are concatenated with optional user-uploaded documents. The combined context is passed to an internal GPT-4 turbo model for response generation. Model parameters (temperature, top-p) are user-configurable.

Cloud Infrastructure and Security. The system runs on Amazon EC2 instances with Tesla T4 GPUs. All components are secured within the internal network, protected by Cloudflare Gateway, and require enterprise authentication. Data transmission and storage follow strict encryption protocols, ensuring full compliance with internal governance and data protection standards. An overview of the full architecture is illustrated in Figure 2.

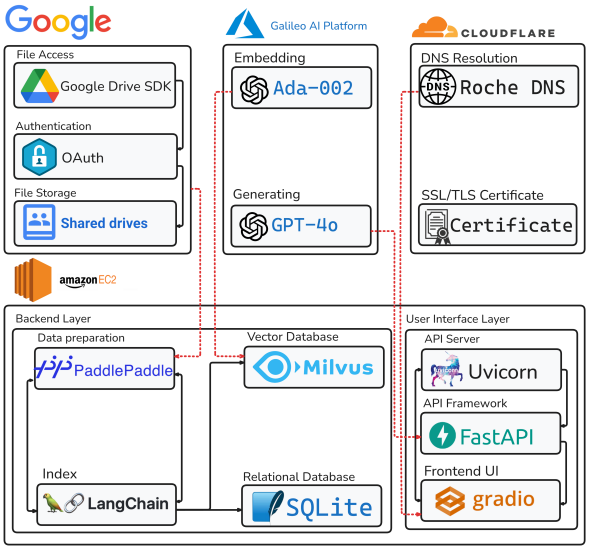

Figure 2: System Architecture Overview for the Regulatory Alignment Guide System

5.2 Evaluation Dataset Construction

To systematically evaluate the performance of different RAG strategies in regulatory question answering, we constructed a structured evaluation dataset consisting of diverse document types with representative question formats.

Stratified Sampling and Preprocessing. Source documents were collected from the internal cloud repository. To ensure coverage and reduce sampling bias, a stratified sampling strategy based on file types was applied. All sampled documents were processed via an automated parsing pipeline that handled format normalization, encoding compatibility, and structural cleaning. This resulted in a unified textual corpus suitable for downstream QA generation.

Automatic QA Pair Generation. We employed the GPT-4 model via an internal **Galileo AI Platform** to generate question-answer (QA) pairs from the cleaned documents. Prompt templates were designed to guide the model in extracting key information and generating well-formed questions, concise answers, and supporting evidence from the source text. Each QA instance was linked to the corresponding document metadata for full traceability and interpretability[5].

Dataset Structure. The final dataset is serialized in JSON format, where each entry contains:

- `file_name`: Source document identifier;
- `question`: A natural language query representing realistic regulatory review tasks;
- `answer`: The GPT-4 generated response, used as a system performance reference;

[3] See Appendix B for monitoring statistics over 30 days.
[4] See Appendix C, Figure 6 for the full interface layout.
[5] See Appendix D for prompt details.

- `answer_source`: The original document span supporting the answer, used for grounding and relevance evaluation.

5.3 Fine-tuning Dataset Construction

To train a task-specific reranker for regulatory question answering, we constructed a binary-labeled dataset tailored for passage-level semantic relevance modeling. This dataset forms the supervision signal for cross-encoder fine-tuning in the reranking module.

Document Sampling and QA Generation. Using a stratified sample of internal regulatory documents (PDF, Word, Excel), we extracted text content through an automated pipeline and invoked the GPT-4 model via the **Galileo AI Platform** to generate initial QA pairs. Prompts were carefully designed to elicit logically structured, verifiable QA examples, each grounded in specific document spans.

Positive Instance Collection. Each generated QA pair was recorded with the associated file metadata and the supporting text passage. Only high-quality pairs with non-empty answers and explicit supporting evidence were retained as positive samples, each labeled with `"label": 1`.

Negative Sampling Strategy. For each positive instance, we applied a multi-stage negative sampling strategy. Negatives were selected from unrelated passages across other documents (cross-document), semantically distinct segments from the same document (intra-document), or fallback segments when needed. Each negative instance reuses the same question as the positive sample, but pairs it with a semantically irrelevant passage, labeled with `"label": 0`.

Dataset Format. The final dataset is serialized in JSONL format, with each line representing one training instance. Each entry contains:

- `question`: The regulatory query;
- `passage`: The candidate passage (positive or negative);
- `label`: A binary label indicating relevance (1 for positive, 0 for negative);
- `file_name` / `file_id`: Metadata for traceability;
- (Optional) `answer`, `answe_source`: Available only for positive samples, used for evaluation alignment.

5.4 Results and Analysis

We conduct extensive experiments under various top-K retrieval settings ($K \in \{3, 5, 10, 15\}$) to systematically evaluate the effectiveness of our proposed HiSACC chunking method and BGE-Reranker, a fine-tuned reranking model based on the BGE architecture, in the context of regulatory question answering. Specifically, we assess four configurations: Recursive Chunking (RC), HiSACC, and their respective variants enhanced with BGE-Reranker. The evaluation is carried out using nine metrics that capture retrieval quality, grounding accuracy, and fluency, as introduced in Section 4.4. The results are summarized in Table 1.

Main Results: HiSACC+BGE-Reranker consistently outperforms all baselines. Across all values of K, the combined system of HiSACC and BGE-Reranker achieves top scores on at least 8 out of 10 evaluation metrics, showing consistent superiority over alternative configurations. At $K = 15$, it reaches the best performance on FT (Faithfulness Test, 0.9252), LF (Language Fluency, 0.8648), GR (Groundedness Rate, 0.8453), and achieves the lowest ORP (Over-Retrieval Penalty, 0.0054). These results demonstrate that integrating hierarchical chunking with domain-tuned reranking leads to precise, fluent, and verifiable answers, especially critical in high-stakes regulatory compliance scenarios.

HiSACC vs. Recursive Chunking: hierarchical segmentation improves retrieval quality and reduces noise. HiSACC outperforms Recursive Chunking at every value of K, regardless of whether reranking is used. It enhances AR, CR, and FIM while reducing ORP. For instance, at $K = 5$, HiSACC increases AR by +1.1 (0.8629 vs. 0.8511), improves FIM by +9.3 points (0.7526 vs. 0.6594), and lowers ORP from 0.0062 to 0.0041. These gains suggest that HiSACC produces more semantically coherent retrieval segments, which not only match user intent more precisely but also reduce the inclusion of irrelevant or fragmented context.

Effect of BGE-Reranker: post-retrieval reranking enhances grounding and factual alignment. The BGE-Reranker module delivers consistent improvements across both chunking methods. At $K = 10$, for example, Recursive Chunking with reranker improves FT from 0.8700 to 0.9082, and ASM from 0.9147 to 0.9304. Similar gains are observed with HiSACC, where BGE enhances GR from 0.8092 to 0.8320 and raises AR from

K	Configuration	AR	CR	GR	FIM	CC	ASM	LF	ORP↓	FT
3	Recursive Chunking	0.865312	0.864210	0.823415	0.688100	0.885900	0.924200	0.850300	0.005200	0.890120
	HiSACC	**0.873452**	**0.874911**	**0.830737**	**0.762411**	**0.891732**	0.930871	0.853912	0.004100	0.902345
	Recursive Chunking BGE	0.864199	0.867011	0.825982	0.728411	0.879814	0.928012	0.849281	0.004600	0.896541
	HiSACC BGE	0.878641	0.872810	0.835924	0.755890	0.894921	**0.934512**	**0.860214**	**0.003800**	**0.910983**
5	Recursive Chunking	0.851112	0.853488	0.812925	0.659431	0.872447	0.916089	0.835240	0.006237	0.884201
	HiSACC	**0.862901**	**0.870215**	**0.829446**	**0.752603**	**0.889975**	0.924871	0.838899	0.004100	0.902712
	Recursive Chunking BGE	0.855297	0.859811	0.820144	0.723882	0.880412	**0.926381**	**0.845601**	0.004530	0.894972
	HiSACC BGE	0.868541	0.868812	0.834919	0.757811	0.892970	0.930042	0.848231	**0.003480**	**0.912870**
10	Recursive Chunking	0.842177	0.842935	0.799814	0.621990	0.861004	0.914672	0.821104	0.007131	0.870003
	HiSACC	0.845902	0.846711	0.809235	0.735112	0.881305	0.921841	0.835244	0.003920	0.901882
	Recursive Chunking BGE	**0.860711**	0.860992	**0.829777**	0.752004	0.891104	0.930421	**0.840713**	0.003510	0.908173
	HiSACC BGE	0.857981	**0.862511**	0.831962	**0.769112**	**0.893211**	**0.932481**	0.837102	**0.003250**	**0.914903**
15	Recursive Chunking	0.831211	0.832164	0.789201	0.602741	0.850712	0.911114	0.809411	0.008021	0.852141
	HiSACC	0.855442	0.860314	0.818912	0.741218	0.882014	0.922878	0.839911	0.006912	0.888441
	Recursive Chunking BGE	**0.870711**	0.875112	0.837742	0.781414	0.892441	0.935211	0.859901	0.006101	0.915674
	HiSACC BGE	0.872901	**0.878921**	**0.845334**	**0.803924**	**0.901015**	**0.940112**	**0.864771**	**0.005423**	**0.925204**

Table 1: Metric Results for different configurations and K values.

0.8459 to 0.8580. This highlights the reranker's effectiveness in eliminating distractor chunks that may be superficially similar but lack semantic alignment with the actual query intent, leading to more grounded and contextually supported answers.

6 Conclusion and Future Work

This research presents a robust AI-powered regulatory compliance assistant that leverages hierarchical semantic chunking (HiSACC) and domain-adaptive reranking (BGE-Reranker) within a Retrieval-Augmented Generation (RAG) architecture. Through comprehensive evaluation across multiple metrics—including groundedness, answer relevance, and over-retrieval penalty—the system demonstrates significant improvements in contextual accuracy, factual alignment, and language fluency over traditional baseline methods. The consistent performance across varying retrieval depths (K) affirms the scalability and reliability of the proposed methods in real-world pharmaceutical compliance scenarios.

Beyond technical contributions, this work offers a practical pathway toward reducing manual overhead and mitigating regulatory risk in high-stakes environments. By encoding enterprise knowledge into dynamically retrieved, contextually grounded responses, the system bridges the gap between complex, evolving regulations and actionable compliance decisions.

Looking ahead, two critical directions can further advance the intelligence and autonomy of the system: the integration of a Model Context Protocol (MCP) and Reinforcement Learning from Human Feedback (RLHF). MCP will enable LLMs to access distributed internal and external regulatory data sources through a unified, plugin-based interface—drastically reducing manual ingestion bottlenecks and enabling real-time compliance alignment. Meanwhile, RLHF introduces a feedback loop between compliance officers and the model, allowing the system to iteratively learn from domain-specific user preferences and factual corrections. This transition from static prompting to continuous human-in-the-loop optimization will improve both the quality and auditability of system outputs.

Together, these future enhancements pave the way toward a more autonomous, adaptive, and regulation-aware AI framework—not only for pharmaceuticals, but also for other compliance-critical industries such as finance, data privacy, and cybersecurity.

Limitations

Despite promising performance in regulatory question answering and compliance-aware generation, our system presents several limitations in architecture design, user interaction, scalability, and knowledge updating capabilities.

Ephemeral Context Fusion. The system employs a dual-source context mechanism, integrating both persistent semantic retrieval from Milvus and real-time content uploaded via the frontend. While this design allows dynamic fusion of long-term and ad-hoc contexts, uploaded documents are treated as transient input and are not persisted for future retrieval. This limits support for multi-turn interaction, user-specific memory, and longitudinal compliance tracking.

Shallow Context Integration. The current strategy concatenates retrieved and uploaded content without modeling semantic hierarchy or salience.

This naive fusion may introduce context conflicts in multi-document scenarios and lack interpretability. Additionally, due to the token limits of large language models (e.g., GPT-4o), only one document can be uploaded per query, constraining the system's applicability to long-form reporting or comparative analysis.

Frontend Usability Constraints. The frontend interface is built with Gradio, which facilitates rapid deployment but lacks extensibility for enterprise-level use. Essential features—such as multilingual support, persistent session memory, interactive chunk highlighting, and advanced input modalities (e.g., voice or structured queries)—are not currently supported. This reduces usability in professional auditing or multilingual regulatory review scenarios.

Concurrency Fragility. The backend does not currently implement robust throttling or fallback mechanisms. Under high concurrency, API requests to the language model may exceed rate limits, leading to request failures without queuing, retry, or graceful degradation strategies. This limits system reliability in production or peak usage environments.

Incomplete Knowledge Synchronization. Although the backend supports periodic synchronization from Google Shared Drives, real-time updates are constrained by the manual maintenance of source documents. The system is not yet integrated with structured compliance databases or internal regulatory platforms, reducing responsiveness in rapidly changing regulatory contexts.

References

Tom B. Brown et al. 2020. Language models are few-shot learners. *OpenAI Blog*. Accessed: 2024-11-12.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, pages 129–136.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*.

ComplianceQuest. 2025. Regulatory compliance for pharmaceutical industry. Blog post, ComplianceQuest.

André V. Duarte et al. 2024. Lumberchunker: Long-form narrative document segmentation. *Journal of Document Analysis*, 29(1):1–34.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2025. Ragas: Automated evaluation of retrieval augmented generation.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Journal of AI Research*.

Google Developers. 2024a. About the google drive api | google drive | google for developers.

Google Developers. 2024b. Manage downloads | google drive | google for developers.

Zhiqing Ji, Nayeon Lee, Rita Frieske, Tongshu Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys (CSUR)*, 55(12):1–38.

Ziwei Ji, Nayeon Lee, et al. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*. Accessed: 2024-11-12.

Yun Jiang, Zilong Xie, Wei Zhang, Yun Fang, and Shuai Pan. 2024. E2e-afg: An end-to-end model with adaptive filtering for retrieval-augmented generation. *Advanced Institute of Information Technology, Peking University*. https://github.com/XieZilongAI/E2E-AFG.

Sandra Jordan, Simona Sternad Zabukovšek, and Irena Šišovska Klančnik. 2022. Document management system–a way to digital transformation. *Naše gospodarstvo/Our economy*, 68(2):43–54.

Nandkumar S. Kher. 2020. Demystifying regulatory compliance in the pharmaceutical industry. *Journal of Pharmaceutical Care & Health Systems*, 7(4):1–2.

Seungyoon Kim and Seungone Kim. 2024. Can language models evaluate human written text? case study on korean student writing for education.

Hamin Koo, Minseon Kim, and Sung Ju Hwang. 2024. Optimizing query generation for enhanced document retrieval in rag. *arXiv preprint arXiv:2407.12325*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization.

David Rau, Shuai Wang, Hervé Déjean, and Stéphane Clinchant. 2024. Context embeddings for efficient answer generation in rag. *ACM Transactions on Information Systems*, 42(4):1–13.

Akash Sharma, Vriti Gamta, and Gaurav Luthra. 2023. Regulatory compliance in the united states: A comprehensive analysis of usfda guidelines and implementation strategies. *Journal of Pharmaceutical Research International*, 35(17):41–50.

Kaize Shi, Xueyao Sun, Qing Li, and Guandong Xu. 2024. Compressing long context for enhancing rag with amr-based concept distillation. *University of Technology Sydney and The Hong Kong Polytechnic University*. Preprint. Under review.

U.S. Food and Drug Administration. 2024. Code of federal regulations: Title 21, part 211—current good manufacturing practice for finished pharmaceuticals. Federal Register.

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, et al. 2024a. Searching for best practices in retrieval-augmented generation. *arXiv preprint arXiv:2407.01219*.

Yujing Wang, Hainan Zhang, Liang Pang, Hongwei Zheng, and Zhiming Zheng. 2024b. Maferw: Query rewriting with multi-aspect feedbacks for retrieval-augmented large language models. *arXiv preprint arXiv:2408.17072*.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, et al. 2022. Finetuned language models are zero-shot learners. *ICLR*. Accessed: 2024-11-12.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding.

Shi Yu et al. 2024. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *Journal of AI Research*, 58(3):657–690.

Jihao Zhao et al. 2024. Meta-chunking: Learning efficient text segmentation via logical perception. *Journal of Computational Linguistics*, 50(2):205–240.

A Corpus Construction and Preprocessing

All documents used in this study were collected from a Google Shared Drive. File-level metadata—including MIME type, name, path, ID, and modification timestamp—was extracted to support structured processing. The initial MIME-type distribution is shown in Table 2.

File Type	MIME Type	Count
PDF File	application/pdf	454
Word Document (`.docx`)	application/vnd.openxmlformats-officedocument.wordprocessingml.document	169
Microsoft Word (`.doc`)	application/msword	44
Excel File (`.xlsx`)	application/vnd.openxmlformats-officedocument.spreadsheetml.sheet	12
Google Sheets	application/vnd.google-apps.spreadsheet	4
Google Docs	application/vnd.google-apps.document	2
URL File	text/x-url	2
Google Apps Script File	application/vnd.google-apps.script	1
Excel File (`.xls`)	application/vnd.ms-excel	1
Presentation File	application/vnd.openxmlformats-officedocument.presentationml.presentation	1

Table 2: Initial distribution of file MIME types.

Files unsuitable for text extraction (e.g., URL and script files) were removed. Table 3 shows the retained document types used in downstream processing.

File Type	MIME Type	Count
PDF File	application/pdf	454
Word Document (`.docx`)	application/vnd.openxmlformats-officedocument.wordprocessingml.document	169
Microsoft Word (`.doc`)	application/msword	44
Excel File (`.xlsx`)	application/vnd.openxmlformats-officedocument.spreadsheetml.sheet	12
Google Sheets	application/vnd.google-apps.spreadsheet	4
Google Docs	application/vnd.google-apps.document	2
Excel File (`.xls`)	application/vnd.ms-excel	1

Table 3: Retained file types after filtering out non-extractable formats.

Text extraction was conducted using format-specific parsers. OCR was applied to scanned PDFs. The overall text extraction success rate exceeded 90%. Table 4 summarizes failure statistics before and after OCR.

File Type	Initial Failures	Post-OCR Failures
Word Document (`.docx`)	2	2
Excel File (`.xlsx`)	1	1
Google Sheets	3	3
Excel File (`.xls`)	1	1
PDF File	140	5

Table 4: Text extraction failures before and after OCR processing.

Unprocessable files (e.g., encrypted or empty documents) were logged and excluded from the experimental pipeline.

B Google Drive API Usage Monitoring

To evaluate the reliability and performance of our document ingestion pipeline, we monitored API usage across a 30-day window. As shown in Figures 3–5, the Google Drive API exhibited stable, low-frequency traffic, with occasional spikes corresponding to bulk synchronization events. Error analysis (Figure 4) reveals that while the Drive API maintained a manageable 5% error rate across 7,858 requests, the Gemini for Google Cloud API experienced consistent failure (100% error rate across 42 requests), indicating critical service incompatibility. In terms of latency (Figure 5), the Drive API showed a median response time of 163 ms, with outliers reaching up to 2.8 seconds. These observations confirm that the Google Drive API provides a relatively stable and scalable foundation for downstream document alignment tasks.

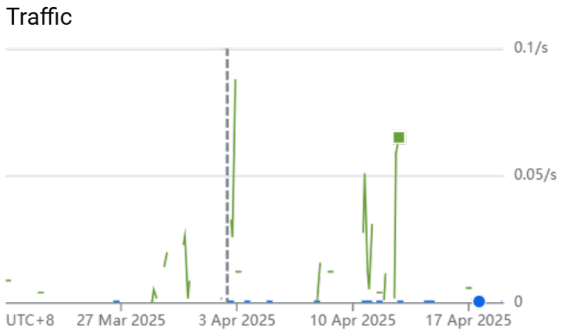

Figure 3: Traffic rate of the Google Drive API over 30 days.

C User Interface Overview

The system front-end is implemented using Gradio, providing an intuitive interface for user interaction. Users can input natural language queries, optionally enable Milvus-based retrieval, and upload documents for context-specific analysis. The

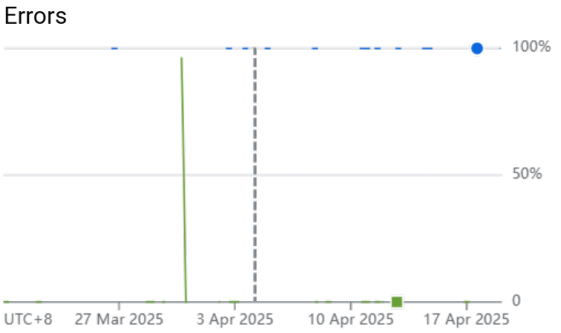

Figure 4: Error rates of Drive and Gemini APIs.

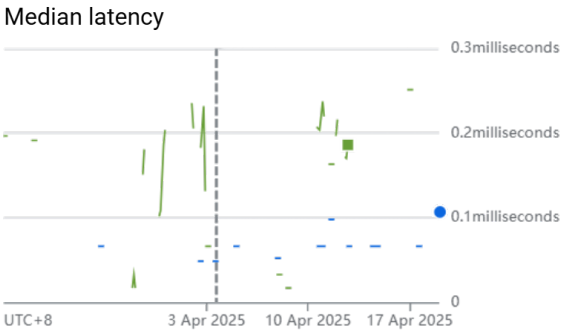

Figure 5: Median latency trends of API responses.

interface also includes adjustable parameters under "Advanced Settings" (specifically, temperature and top-p) to control the response behavior of the language model. A feedback section is available to collect user comments, facilitating potential system improvement.

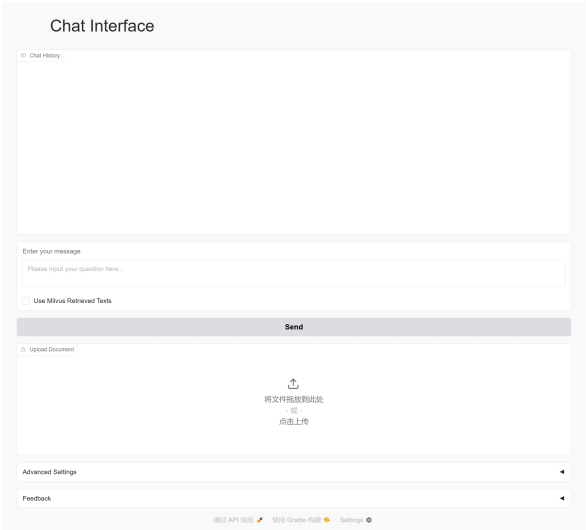

Figure 6: Gradio-based front-end interface

D Prompt Design for QA Generation Task

Prompt for Generating Question-Answer Pairs from a Document

In the following task, you are given a complete semi-structured or structured document. You are a senior document analysis expert familiar with medical, legal, and standardized documents. Your task is to generate high-quality question-answer pairs from the document. These pairs will be used to assess the comprehension ability of semantic retrieval and generation models in a Retrieval-Augmented Generation (RAG) system.
You should proactively identify high-value information that can be questioned and produce multiple challenging and practically meaningful Q&A pairs. Questions should avoid simple paraphrasing or copying, and instead focus on logic, synthesis, or judgment. Answers should be concise and accurate, supported by a direct sentence from the document.
Output format (JSON array):

```
[
  {
    "question": "Question content",
    "answer": "Concise answer",
    "answer_source": "Sentence from the document that supports the answer"
  },
  ...
]
```

Requirements:

- Include factual, procedural, comparative, or reasoning-based questions.
- Answers must be verifiable. Subjective speculation is not allowed.
- The answer_source must be a direct quote from the document that supports the answer.

Task steps:

1. Read the entire document to understand its topic and structure.
2. Identify valuable questions beyond surface-level content.
3. Provide concise and accurate answers.
4. Quote directly from the document as the answer_source.
5. Return the output as a JSON array.