ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

School of Information Sciences and Technology
Department of Informatics
Athens, Greece

Bachelor Thesis
in
Computer Science

Retrieval Augmented Generation on Regulatory Documents

Ioannis Chasandras

Supervisor: Prof. Ion Androutsopoulos
Co-supervisor: Odysseas Chlapanis

June 2025

Ioannis Chasandras

Retrieval Augmented Generation on Regulatory Documents

June 2025

Supervisor: Prof. Ion Androutsopoulos

Athens University of Economics and Business

School of Information Sciences and Technology

Department of Informatics

Information Processing Laboratory, Natural Language Processing Group

Athens, Greece

Abstract

This thesis investigates the application of Retrieval Augmented Generation (RAG) in regulatory procedures through the emerging field of Regulatory NLP. Based on real-world regulatory documents, the study evaluates the performance of commercial retrieval models and introduces advanced, hybrid retrieval techniques tailored for legal compliance tasks. Given the critical need for precision and completeness in the legal domain, new algorithms that utilize Large Language Models (LLMs) are developed to enhance regulatory question-answering. The work also includes an adversarial evaluation of RePASs, a metric focused on legal obligations. Through participation in the RIRAG-2025 shared task, the thesis demonstrates both the promise and current limitations of AI systems in regulatory settings, emphasizing the need for further exploration in this field.

Περίληψη

Η παρούσα πτυχιακή εργασία εξερευνά την εφαρμογή της τεχνολογίας Retrieval Augmented Generation (RAG) σε διαδικασίες που αφορούν κανονισμούς στο ανερχόμενο πεδίο της Επεξεργασίας Φυσικής Γλώσσας για Ρυθμιστικά κείμενα (Regulatory NLP). Βασιζόμενη σε πραγματικά έγγραφα κανονισμών, η μελέτη αυτή αξιολογεί την επίδοση εμπορικών μοντέλων ανάκτησης πληροφοριών και εισάγει προηγμένες, υβριδικές τεχνικές ανάκτησης πληροφοριών ειδικά προσαρμοσμένες για εργασίες νομικής συμμόρφωσης. Δεδομένης της κρίσιμης ανάγκης για ακρίβεια και πληρότητα στο νομικό πεδίο, νέοι αλγόριθμοι αναπτύχθηκαν οι οποίοι αξιοποιούν μεγάλα γλωσσικά μοντέλα (LLMs) για την ενίσχυση των απαντήσεων σε ρυθμιστικές ερωτήσεις. Η εργασία περιλαμβάνει την αξιολόγηση του RePASs, μίας μετρικής που επικεντρώνεται σε νομικές υποχρεώσεις. Μέσω της συμμετοχής στον σχετικό διεθνή διαγωνισμό RIRAG 2025, η συγκεκριμένη πτυχιακή εργασία αναδεικνύει τόσο τις υποσχέσεις όσο και τους περιορισμούς των συστημάτων Τεχνητής Νοημοσύνης σε ρυθμιστικά περιβάλλοντα, τονίζοντας την ανάγκη για περαιτέρω έρευνα στο εν λόγω πεδίο.

Acknowledgements

I would like to express my respect and gratitude to my professor and supervisor, Ion Androutsopoulos, for granting me the opportunity to work alongside him in this fascinating research field. His mentorship has been instrumental in shaping my academic development and research perspective. I am also thankful to Ph.D. candidate Odysseas Chlapanis for his invaluable assistance, constructive feedback and kind advice, which helped me navigate complex concepts with greater clarity. Finally, I am grateful to my family and friends for their love, patience, and belief in me.

Contents

1 Introduction

The increasing complexity and volume of regulatory requirements have made legal compliance a significant and growing challenge for organizations. Regulatory NLP (RegNLP) [Goa+23] has emerged as a field focused on applying Natural Language Processing techniques to help interpret, manage, and organize regulatory texts. Ensuring compliance with regulations often requires significant time, expertise, and resources. Between 1980 and 2020, the U.S. public and private sectors collectively spent an estimated approximately 292 billion hours adhering to more than 36,000 regulations, representing roughly 3.2% of total annual working hours [Kal23]. At the same time, compliance failures can result in severe financial penalties. For example, FINTRAC recently imposed a 7.5 million Canadian dollar fine on the Royal Bank of Canada for not reporting suspicious transactions.[1] These examples highlight the urgent need for more efficient ways for experts to access relevant regulatory documents and evaluate compliance with applicable regulations. The main focus of this thesis is Retrieval-Augmented Generation (RAG), which offers a promising solution by combining information retrieval and natural language generation. In this approach, given a question, a retrieval system first identifies and extracts the most relevant regulatory passages, which are then used by a generation model to produce an accurate, context-aware answer to the question. This not only reduces the time and effort required for manual document review, but also helps ensure that responses are grounded in the actual regulatory text, thus improving both efficiency and reliability in compliance tasks.

In this study, we investigate a range of retrieval methods aimed at identifying relevant passages from regulatory documents in response to a given question. We evaluate the performance of commercially available neural retrievers that have promised high-quality results in domains related to ours. In addition, we incorporate lexical retrieval approaches and explore hybrid retrieval strategies that combine multiple methods to better adapt to domain-specific requirements. For the generation component, we implement a set of algorithms designed to produce accurate and well-grounded responses based on the retrieved content. Furthermore, we conduct an adversarial examination of RePASs [Gok+24], a reference-free and model-based metric, to assess its suitability for evaluating generated responses in the context of regulatory compliance.

The work carried out in this thesis was based on the participation of the AUEB NLP Group in the Regulatory Information Retrieval and Answer Generation shared task (RIRAG-2025)

[1] https://www.bloomberg.com/news/articles/2023-12-05/rbc-hit-with-fine-for-breaking-canadian-money-laundering-rules

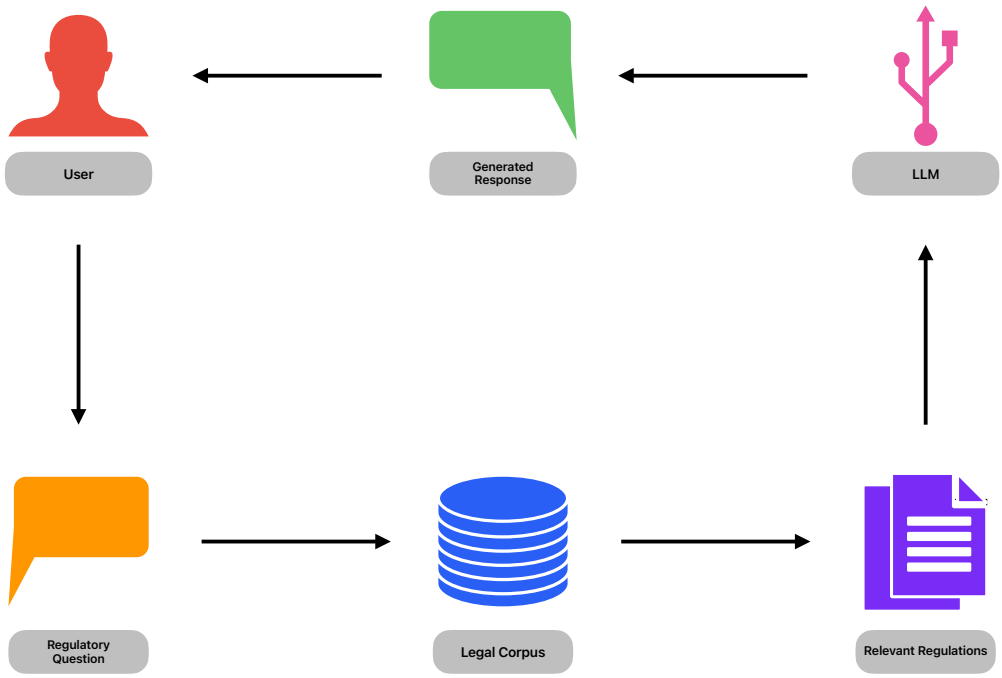

Fig. 1.1: RAG workflow diagram

[Gok+25], which was part of the RegNLP Workshop that took place in January 2025 in conjunction with COLING 2025.[2][3] The competition focused on improving regulatory compliance through intelligent information retrieval and answer generation. The task is based on the ObliQA (Obligation Question Answer) dataset, consisting of compliance-related questions and associated regulatory texts from Abu Dhabi Global Markets (ADGM).[4] Participants were challenged to build systems capable of accurately retrieving relevant legal passages and generating precise, coherent answers. The task included two subtasks: Passage Retrieval and Answer Generation. The first focused on identifying relevant obligations and rules from regulatory documents in response to a question, while the second required the synthesis of this information into clear, accurate answers. We submitted three systems that secured the 4th place in the Passage Retrieval subtask and the 3rd place in the Answer Generation subtask among 20 competitors. The code developed for this participation is publicly released.[5]

1.1 Motivation and Problem Statement

The complexity and high stakes of regulatory compliance demand tools that ensure accurate, complete, and consistent interpretation of large, complex legal texts. RAG provides a

[2] http://nlp.cs.aueb.gr
[3] https://coling2025.org/
[4] https://www.adgm.com/
[5] https://github.com/nlpaueb/verify-refine-repass

promising framework by retrieving relevant regulatory passages and generating context-aware answers. However, standard evaluation metrics often fall short in capturing the strict precision and obligation coverage required in legal settings. This thesis tackles two issues: improving RAG systems for regulatory question answering (through RIRAG-2025) and evaluating RePASs, a reference-free, model-based metric designed to assess grounding, contradiction, and completeness in generated compliance-related answers. The main purpose of this study is to help advance the development of more robust compliance support systems by exploring both system design and evaluation in RegNLP.

1.2 Thesis Structure

Chapter 2: Background and Related Work

This chapter provides a short introduction to the field of Regulatory NLP and an overview of existing RAG pipelines and technologies. It also reviews related work on iterative improvement and adversarial attacks.

Chapter 3: RIRAG 2025

In Chapter 3, a more detailed overview of RIRAG-2025 is presented. An analysis of the ObliQA dataset is conducted, including its structure, differences among documents, and unique characteristics. An explanation of the RePASs metric, which was introduced along with the dataset, is also provided.

Chapter 4: Implemented methods and systems

This chapter is dedicated to describing the methods and systems that were implemented during this thesis. It addresses the retrieval approaches that were tested, as well as the generation algorithms and their variations.

Chapter 5: Experiments

Chapter 5 presents the experimental setup used to evaluate the proposed systems. It describes the evaluation measures and results for each component of the pipeline: retrieval, pre-processing, and generation. The chapter provides insights into the performance of different configurations and methods.

Chapter 6: Conclusions

The final chapter summarizes the main methods and findings of this thesis. It also highlights the limitations of reference-free metrics and proposes directions for future work.

2 Background and Related Work

2.1 Background

2.1.1 Regulatory NLP

Regulatory NLP (RegNLP) is a new subfield of legal artificial intelligence and natural language processing introduced by Goanta et al. [Goa+23], whose aim is to tackle the challenges arising at the intersection of NLP technologies and regulatory processes. The main topic of RegNLP is the investigation of NLP applications in regulatory documents and the development of systems that can assist regulatory processes in the service of public interest, legal compliance, and innovation. This includes tasks such as extracting obligations, mapping risk factors, aligning automated systems with legal frameworks, and supporting policy analysis.

The development of RegNLP reflects a broader shift in NLP research toward proactively shaping policy and regulation, rather than passively responding to the risks of new technologies. Although the NLP community has made significant progress in identifying and mitigating technical harms, it has often done so without integrating insights from regulatory studies, which offer well-established methods for assessing and managing risk and uncertainty in the context of innovation [Goa+23].

At the same time, regulatory bodies are under increasing pressure to respond to rapid technological change, a challenge often described in the literature as the 'pacing gap' between the speed of innovation and the slower evolution of legal and regulatory responses [Goa+23]. In this context, RegNLP holds promise as a way to bridge this gap by providing tools and insights that allow regulators to access and interpret technical knowledge ad hoc, effectively [Goa+23].

Goanta et al. [Goa+23] set the ultimate aim of RegNLP not only to support the regulation of AI and NLP technologies, but also to enable NLP as a field to contribute more meaningfully to broader governance and regulatory processes.

2.1.2 Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) is a framework that combines large language models with external knowledge sources to improve the factual accuracy and address some of the limitations of purely parametric models, such as hallucinations and static knowledge [Lew+20]. By integrating a nonparametric memory, like a document index, with a generative language model, RAG enables systems to retrieve and incorporate relevant information at inference time. This approach has proven especially effective in knowledge-intensive tasks where up-to-date or domain-specific information is needed [Gao+24]. Given the dynamic and complex nature of regulatory documents, RAG's ability to supplement the generation with reliable external evidence and also keep up with new regulations by updating its external knowledge base efficiently makes it a compelling fit for the tasks of RegNLP.

Retrieval is a key step in RAG, where the system identifies relevant text chunks from a large collection of documents based on the user query. The retrieval step builds on a variety of traditional and modern information retrieval techniques, with vector-based methods and neural embeddings now dominating the field. This typically involves representing both the query and the documents as vectors using the same embedding model and computing semantic similarity scores between them [Gao+24]. The top ranked chunks are selected as contextual evidence and passed along to the generation module. Models like Dense Passage Retriever (DPR) are often used for this purpose due to their effectiveness in dense retrieval scenarios [Lew+20]. In the context of RegNLP, this retrieval component can be customized to navigate large legal corpora, making it easier to locate precise legislative or policy-relevant content.

Generation in RAG takes place after retrieval, where the language model is tasked with producing a response based on both the query and the retrieved documents. Depending on the use case, the system may generate responses solely from the retrieved content or integrate this information with its own parametric knowledge [Gao+24]. This flexibility is particularly useful for generating explanations grounded in regulatory texts, which often require both precision and contextual understanding.

Evaluation of RAG systems typically focuses on performance across knowledge-intensive tasks such as open-domain question answering or fact verification. Usually, the evaluation of the retrieval and generation components of a RAG system can be evaluated separately. Experiments have shown that RAG models achieve state-of-the-art results on benchmarks like Natural Questions [Kwi+19] and WebQuestions [Ber+13], outperforming traditional extractive methods even in tasks not originally designed for generation [Lew+20].

2.2 Related Work

2.2.1 Keyword-Based Lexical Retrieval with BM25

Okapi BM25 [Rob+95] is a widely used Bag-of-Words ranking function in information retrieval that estimates the relevance of a document to a query based on term frequency and document statistics. BM25 addresses two key limitations of earlier models: (1) accounting for the frequency of query terms in documents, and (2) compensating for variation in document lengths in full-text corpora.

The BM25 scoring function is based on the idea that the relevance of a document increases with the frequency of a query term, but with diminishing returns, and that longer documents should be normalized to prevent length bias [MRS08; Rob+95]. Following the formulation used in the Okapi system developed by Robertson et al. [Rob+95], the BM25 score of a document d with respect to a query Q is calculated as:

$$\text{BM25score}(d, Q) = \sum_{t \in Q} \mathit{idf}(t) \cdot \frac{\mathit{tf}(t) \cdot (k_1 + 1)}{\mathit{tf}(t) + k_1 \cdot \left(1 - b + b \cdot \frac{\mathit{dl}(d)}{\mathit{avdl}}\right)} \tag{2.1}$$

where:

- $\mathit{tf}(t)$ is the frequency of term t in document d,
- $\mathit{dl}(d)$ is the length of document d,
- *avdl* is the average document length in the collection,
- k_1 is a positive constant that controls term frequency saturation,
- $b \in [0, 1]$ controls the strength of document length normalization.

The inverse document frequency (IDF) component is defined as:

$$\mathit{idf}(t) = \log\left(\frac{N - n_t + 0.5}{n_t + 0.5} + 1\right) \tag{2.2}$$

where N is the total number of documents in the collection and n_t is the number of documents containing the term t. This IDF formulation is rooted in the Robertson–Sparck-Jones probabilistic model and provides an effective estimate of term informativeness in the absence of relevance feedback [Rob+95]. It increases the influence of rarer terms while suppressing common terms that heavily influence ranking.

In essence, BM25 strikes a balance between empirical effectiveness and computational efficiency. It remains a cornerstone of classical information retrieval, widely adopted in both academic and industrial settings, and frequently used as a strong non-neural baseline for retrieval tasks [MRS08].

2.2.2 Dense Semantic Retrieval

Traditional information retrieval systems, such as BM25, rely on sparse bag-of-words representations to rank documents based on lexical overlap with a query. Although these methods are efficient and interpretable, they often fail to capture semantic similarity between words or phrases. This limitation motivated the development of dense retrieval, where both queries and documents are embedded in a continuous multidimensional vector space and similarity is calculated using a fixed metric such as cosine similarity. Dense retrieval allows for better generalization by capturing the meaning of words beyond their surface forms.

One of the earliest works in dense retrieval is ORQA (Open-Retrieval Question Answering) by Lee et al. [LCT19], which introduced a fully neural retriever trained using an Inverse Cloze Task. In ORQA, each question and document is encoded using a BERT-based encoder, and relevance is measured via the dot product between their embeddings. This method showed that dense retrievers could outperform BM25 in open-domain QA settings, marking a significant shift in retrieval paradigms. Building on this, Karpukhin et al. [Kar+20] introduced Dense Passage Retrieval (DPR), which simplified the architecture by using two separate encoders (dual encoder) for queries and passages and trained the model using hard negatives. DPR demonstrated strong improvements over traditional retrievers and helped establish dense retrieval as a competitive approach.

Although both ORQA and DPR define how to score query-document pairs, Reimers and Gurevych [RG19] addressed the problem of how to compute sentence embeddings effectively by proposing Sentence-BERT (SBERT). Unlike the original BERT, which is not optimized for producing fixed-size sentence representations, SBERT uses a siamese network to map sentences into dense vectors that can be compared using cosine similarity. This drastically reduces retrieval time while maintaining high accuracy, making it suitable for semantic search, clustering, and question-answering. SBERT proved especially effective in capturing semantic similarity between sentences, which is crucial for meaningful retrieval in many NLP tasks.

Cosine similarity is commonly used to compare dense embeddings in retrieval systems. Given two vectors $\mathbf{u}$ and $\mathbf{v}$, the cosine similarity is defined as

$$\cos(\theta) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|},$$

where $\|\mathbf{u}\|$ is the Euclidean norm of $\mathbf{u}$. When embeddings are normalized, cosine similarity simplifies to the dot product. This metric measures the angle between the two vectors, indicating how similar their directions are in the high-dimensional embedding space. Since dense retrieval is based on semantic closeness rather than lexical overlap, cosine similarity provides an effective and interpretable way to rank documents according to their meaning [RG19; Kar+20].

2.2.3 Hybrid Retrieval

Hybrid retrieval involves multiple retrieval systems and usually combines the strengths of lexical and semantic search to improve document relevance ranking [BGI23]. Traditional lexical methods, such as BM25, rely on exact keyword matching and are practical and systematic in many cases, but suffer from the vocabulary mismatch problem and fail to capture semantic meaning. On the other hand, semantic search leverages pre-trained language models like BERT to embed queries and documents in a dense vector space, allowing for more nuanced similarity based on meaning. However, a known limitation of dense embeddings is their tendency to conflate semantically similar but lexically distinct entities. For example, "I want a flight to Paris" and "I want a flight to London" may be embedded very closely, making it harder to distinguish between specific named entities. Sparse retrieval methods like BM25 are often more effective in capturing exact term matches and entity distinctions. Recognizing that these two approaches offer complementary strengths, recent research has focused on $fusing$ their outputs into a single-ranked list, known as hybrid search [BGI23].

A fusion method that combines the results of two retrieval systems is rank-fusion. This approach works by computing a weighted combination of scores produced by lexical and neural retrievers, such as BM25 and BERT-based models, respectively [BGI23]. One of the earliest and most influential techniques for combining retrieval scores is linear interpolation, introduced by Bartell et al. [BCB94], where the score of each system is scaled and summed according to the assigned weights. In this work, the interpolation approach used by Lin et al. [LNY21] and Wang et al. [WZZ21] is adopted, which apply min-max normalization to ensure that lexical and semantic scores are on a comparable scale. Specifically, the final document score is computed as:

$$s(d) = \alpha \cdot \hat{s}_{\text{BM25}}(d) + (1 - \alpha) \cdot \hat{s}_{\text{neural}}(d), \tag{2.3}$$

where $\alpha \in [0, 1]$ controls the weight between the two systems. The normalized score $\hat{s}$ is calculated as:

$$\hat{s}(d) = \frac{s(d) - s_{\min}}{s_{\max} - s_{\min}} \tag{2.4}$$

where $s(d)$ is the original score of document d, $\hat{s}(d)$ is the normalized score, and $s_{\min}$ and $s_{\max}$ are the minimum and maximum scores, respectively, among all candidate documents for the query. Bruch et al. [BGI23] emphasize the importance of normalization, noting that lexical and semantic scores often have incompatible ranges. Without normalization, a fixed interpolation weight may lead to inconsistent or biased fusion behavior [BGI23].

Wang et al. [WZZ21] explored the effectiveness of score interpolation between BERT-based dense retrievers (DRs) and traditional sparse retrieval models such as BM25. Their empirical analysis shows that while DRs significantly outperform BM25 in shallow evaluation metrics (e.g., MRR@10), they tend to underperform in deep metrics (e.g., MAP, Recall@1000) when used alone. This suggests that BM25 provides weak but complementary lexical signals that dense embeddings often miss and that interpolation can improve retrieval robustness.

Another method that combines the results of several retrieval methods is Reciprocal Rank Fusion (RRF) [CCB09]. RRF is an unsupervised rank aggregation technique that assigns a score to each document based on how highly it is ranked across multiple systems. The score is calculated using the formula:

$$\text{RRFscore}(d) = \sum_{r \in R} \frac{1}{k + r(d)} \tag{2.5}$$

where $r(d)$ is the rank of the document d in the system r, and k is a constant (typically set to 60) used to reduce the dominance of top-ranked documents. This formula ensures that documents ranked highly by at least one system are given priority while still accounting for contributions from other rankings. Despite its simplicity, RRF has been shown to outperform or match more complex fusion methods.

2.2.4 Re-ranking

Re-ranking is a crucial component in modern information retrieval pipelines, particularly in two-stage retrieval systems. After an initial retrieval step returns a candidate set of potentially relevant documents, typically using lexical methods like BM25 or vector-based similarity search, a re-ranker reorders this list to promote the most relevant results to the top [Gao+24]. By focusing only on a small candidate set, re-ranking improves accuracy, acting both as a filter and an enhancer in the retrieval process.

Formally, re-ranking involves taking a list of k documents $R = \{d_1, d_2, ..., d_k\}$ returned by a first-stage retriever and producing a new permutation $R' = \{d'_1, d'_2, ..., d'_k\}$ such that the most relevant documents appear earlier in the list [LNY21]. Since applying expensive transformer-based models to an entire corpus is impractical, re-rankers instead evaluate only the reduced candidate set, computing fine-grained relevance scores between each document and the query. Early re-rankers used rule-based heuristics, but recent advances rely on cross-encoder neural models such as BERT, which jointly encode the query and each document to capture deep semantic interactions [NC20]. Notable examples include BERT-based re-rankers [NC20], Cohere rerank and LLMs like GPT [Gao+24].

Re-rankers offer several advantages over first-stage retrievers. Unlike bi-encoders that separately embed queries and documents, cross-encoder re-rankers process them jointly, allowing the model to attend to complex relationships between words in both texts [Gao+24]. This typically results in much more accurate relevance scoring, especially for subtle or nuanced queries. Moreover, re-rankers are flexible: they can be applied on top of any retrieval system, regardless of whether it is lexical or vector-based, making them a practical and powerful tool for improving retrieval.

2.2.5 Iterative Improvement

Recent advances in LLMs have focused on improving their reasoning accuracy and reliability, particularly for complex tasks. Techniques such as self-refinement, self-consistency, and answer selection have shown significant potential in improving performance without requiring additional training or external supervision.

Self-refinement refers to an iterative process in which the model generates an initial output, evaluates it, and then refines the output based on self-generated feedback. This loop, introduced in the SELF-REFINE framework [Mad+24], enables the model to correct its own mistakes or improve clarity through structured, internal critique. The refinement steps can be repeated several times, and human and automatic evaluations across tasks like dialogue generation, sentiment reversal, and code optimization have demonstrated significant quality improvements. More specifically, the benefits are amplified when the feedback is actionable and specific, allowing the model to effectively target its revisions.

Self-consistency is another powerful approach that departs from single-pass, greedy decoding. Originally introduced by Wang et al. [Wan+23], this method involves generating multiple diverse reasoning paths for a given prompt and then selecting the final answer based on majority voting. The core idea is that complex reasoning problems can be solved in several valid ways, and correct answers tend to appear more frequently across different reasoning trajectories. This marginalization over reasoning paths results in more stable and accurate outputs. For example, combining chain-of-thought prompting with

self-consistency boosted performance on GSM8K by nearly 18%, and showed similar gains across other reasoning datasets like AQuA and SVAMP. Recent extensions of this idea, such as using clustering techniques to estimate model confidence [Wan+24], further enable better calibration and robustness in math reasoning tasks.

In addition, response selection mechanisms play a critical role in verifying and improving generated content. Explanation-Refiner [Qua+24] demonstrates how formal logical feedback from theorem provers can be used to refine natural language explanations iteratively. By converting explanations into formal axioms and using symbolic logic to check validity, the system identifies and removes redundant or logically invalid statements, refining them until a consistent and valid explanation is formed.

Instruction evolution frameworks such as WizardLM [Xu+24] complement these approaches by focusing on the quality and diversity of training data. WizardLM automatically generates increasingly complex instructions using LLMs, evolving tasks to push model limits and improve their ability to follow complex, multi-step instructions. This self-generated, diverse instruction data has been shown to outperform human-written data in terms of downstream model performance, particularly in math and reasoning-heavy benchmarks.

These methods establish a new paradigm in large language model development, emphasizing not only the generation of answers but also their careful refinement, verification, and selection through self-directed and iterative processes. This approach significantly enhances reasoning robustness, calibrates confidence levels, and facilitates improved alignment with user intent across progressively complex tasks.

2.2.6 Adversarial Attacks

Adversarial attacks are techniques that involve subtly modifying inputs to machine learning models in order to trick them into producing incorrect outputs. In the context of deep learning, these perturbations are often minimal and imperceptible to humans, yet can cause models to misclassify data or produce inconsistent results. Such attacks have exposed major vulnerabilities in systems that rely on deep neural networks, especially in safety-critical applications like autonomous driving, biometric security, and language understanding. Common forms of adversarial attacks include evasion attacks, which manipulate inputs during inference time, and poisoning attacks, which tamper with the training data itself. These attacks are studied both to better understand model weaknesses and to develop defense mechanisms that make models more robust to adversarial behavior [Cha+18].

In recent work, two papers applied adversarial attacks for very different purposes in NLP. Huang and Baldwin [HB23] used adversarial attacks to test the robustness of machine translation (MT) evaluation metrics such as BLEURT, COMET, and BERTScore. They used

word-level and character-level attacks to generate minimally perturbed translations that retain the same meaning but receive significantly lower scores from these metrics. They found that these metrics often overpenalize the perturbed sentences and may even behave inconsistently (e.g., BERTScore rating a degraded sentence as semantically similar to the original, but much worse compared to the unperturbed version) [HB23]. Meanwhile, Li et al. [Li+20] proposed BERT-Attack, a method that uses BERT itself as a masked language model to generate fluent and semantically consistent adversarial examples. Their goal was to fool fine-tuned BERT classifiers by replacing key words in the input with context-aware alternatives. They showed that this method had a higher attack success rate and better language quality than previous approaches [Li+20]. These works use adversarial techniques not to "attack" in a malicious sense but to probe the limits of what language models understand and to build more trustworthy systems; the same philosophy was adopted in this thesis, where adversarial examples are used as a tool to test the robustness of RePASs, an evaluation metric explained in Section 3.2 of the following chapter.

3

RIRAG 2025

This chapter provides an overview of the RIRAG 2025 shared task [Gok+25], including the datasets provided by the organizers and utilized in system development. It also presents a detailed explanation of RePASs, a reference-free, model-based evaluation metric. The datasets described here serve as the foundation for experiments throughout this thesis, while RePASs is employed as the primary metric for evaluating generation quality, offering insights into its applicability and effectiveness in regulatory NLP tasks.

3.1 About the shared-task

The Regulatory Information Retrieval and Answer Generation (RIRAG) task was introduced in 2024 to support the development of systems that can understand and respond to complex regulatory questions. It is framed as a two-stage process: first, identifying all relevant obligations from a large collection of regulatory documents, and second, synthesizing a complete and accurate answer based on the retrieved content [Gok+24]. What makes RIRAG especially challenging is the requirement to locate information that may be scattered across multiple documents and then combine that information into a clear response that is concise and comprehensive.

The shared task was officially launched as RIRAG 2025, organized at the RegNLP 2025 workshop to benchmark and compare different approaches [Gok+25]. The task was divided into two subtasks: the first, Information Retrieval, involved returning the most relevant passages for each query, while the second, Answer Generation, required systems to generate answers that reflect all key obligations found in the retrieved passages. To support training and evaluation, participants were provided with the ObliQA dataset, which contains more than 27,000 regulatory questions annotated with their supporting passages, derived from the Abu Dhabi Global Market (ADGM) legal framework. The dataset includes both single-passage and multi-passage questions, encouraging the development of retrieval methods that can handle varying levels of complexity.

3.2 Datasets

The ObliQA dataset [Gok+24] is a regulatory question-answering benchmark that includes 27,869 questions, each paired with one or more relevant regulatory passages.[1] These passages, totaling 13,732 across 40 legal documents provided by ADGM, are predominantly hierarchically structured and contain clause-level obligations, which serve as the grounding material for question generation. The hierarchy of the documents is reflected in the PassageID, capturing their multilevel structure. On average, each document contains 343.3 passages and a mean token count of 24,930.48 tokens, as measured using the NLTK tokenizer [BL04].[2] At the passage level, each passage comprises an average of 2.11 sentences and 72.62 tokens. This hierarchical passage structure, with clause-level granularity and multilevel numbering, enables the development of systems that can retrieve individual passages or aggregate multiple passages for complex questions. ObliQA's design allows for precise evaluation of retrieval accuracy and reasoning, by linking each question to the exact passages that contain the relevant obligations.


```
{
  "ID": "e3515a08-2bd7-4da4-b0ff-9044873943b6",
  "DocumentID": 11,
  "PassageID": "1.1.2",
  "Passage": "Where a Rule prescribes a requirement on a Listed Entity
    or an Undertaking, each Director, Partner or other Person charged
    with the management of that Listed Entity or Undertaking must take
    all reasonable steps within its control to secure compliance with
    the requirement by the Reporting Entity or Undertaking."
}
```


Fig. 3.1: Example passage from the document corpus.


```
{
  "QuestionID": "d34e3516-f053-4652-a0ac-ede703144b9a",
  "Question": "What type of procedures must a Third Party Provider
      establish and maintain to handle issues such as major operational
      and security incidents?",
  "Passages": [
    {
      "DocumentID": 3,
      "PassageID": "20.14.1.(2)",
      "Passage": "As part of that framework, the Third Party Provider
        must establish and maintain effective incident management
        procedures, including for the detection and classification of
        major operational and security incidents."
    }
  ],
  "Group": 1
}
```


Fig. 3.2: Example question from the test set.

[1] https://github.com/RegNLP/ObliQADataset/blob/main/README.md
[2] https://www.nltk.org/

To create this dataset, Gokhan et al. [Gok+24] employed a three-step pipeline involving document curation, synthetic question generation with an LLM, and validation based on natural language inference (NLI). Only passages explicitly expressing regulatory obligations were considered suitable for question generation. For single-passage questions, a direct one-to-one relationship between a question and a clause was preserved. For multi-passage questions, several relevant passages were grouped together using topic-based clustering. After that, passages from the same group were randomly selected to generate questions reflecting the complexity of real-world regulatory queries. Each question, can be answered with 1.32 passages on average and has an mean token count of 26.9.

To ensure the quality of the generated question-passage pair generated, the dataset underwent semantic validation using an NLI model (nli-deberta-v3-xsmall). The model evaluated whether the retrieved passage or passages (premise) entailed the question text (hypothesis), or whether there was contradiction or neutrality. Only pairs that demonstrated entailment or strong alignment were retained. This automated filtering contributed to the overall quality of the dataset, with manual evaluations from ADGM experts confirming that more than 86% of the selected passages were relevant and that nearly 100% of the questions represented realistic and expected compliance queries.

The final dataset includes over 21,000 single-passage questions and more than 6,000 multi-passage questions, with a controlled distribution across training, development, and test sets as seen in Table 3.1. By providing a richly high-quality annotated dataset, ObliQA enables experimentation with a variety of regulatory NLP tasks, including retrieval, generation, summarization, and obligation detection, making it a key resource for advancing automated regulatory compliance systems.

Split	**#Questions**	**1**	**2**	**3**	**4**	**5**	**6**
Train	22295	16946	4016	975	202	100	56
Development	2888	2215	514	116	30	12	1
Test	2786	2126	506	105	36	9	4
Total	27869	21187	5036	1196	268	121	61

Tab. 3.1: Distribution of questions in the ObliQA dataset across training, testing, and development sets, categorized by the number of associated passages.

3.3 RePASs

In vital domains like legal or compliance tasks, accuracy, consistency and completeness are critical characteristics. The Regulatory Passage Answer Stability Score or RePASs for short [Gok+24] is a reference-free evaluation metric that aims to ensure these characteristics by assessing generated responses in regulatory settings.

More specifically, the metric is decomposed into three components: the Entailment Score (E_s), the Contradiction Score (C_s) and the Obligation Coverage Score (OC_s). Each component focuses on a different dimension of answer quality, ensuring the robust evaluation that regulatory texts demand. The Entailment Score examines whether each sentence in the generated answer is based on the content of at least one sentence of the source passages. This component guarantees the accuracy of the response and its factual demands. Contradiction Score quantifies how much the generated answer contradicts the information in the source passage. Contradictory outputs in legal or compliance settings can lead to severe misinterpretations, and thus, minimizing contradiction is critical. The Obligation Coverage Score addresses the dimension of completeness by checking if all regulatory obligations present in the source are mentioned in the answer. This score captures the scope of legal responsibilities ensuring that no mandatory requirement is omitted.

Calculating RePASs consists of three distinct steps that each contribute to evaluating a different dimension of answer quality. In the first step, a sentence-level NLI analysis is conducted between the generated answer and the source passage. Using the cross-encoder/nli-deberta-v3-xsmall model [HGC23], each answer sentence is paired with every sentence in the passage to compute probabilities for entailment, contradiction, and neutrality with the neutral class being discarded. These are organized into two matrices: an entailment matrix and a contradiction matrix. For each answer sentence, the highest entailment and contradiction probabilities across all passage sentences are extracted, corresponding to that sentence. The Entailment Score (E_s) is calculated as the average (over the sentences of the answer) of these maximum entailment scores, and the Contradiction Score (C_s) is similarly derived using the maximum contradiction values. Mathematically, these are defined as follows:

$$E_s = \frac{1}{N} \sum_{i=1}^{N} \max_j P_{\text{entailment}}(p_j, a_i) \tag{3.1}$$

$$C_s = \frac{1}{N} \sum_{i=1}^{N} \max_j P_{\text{contradiction}}(p_j, a_i) \tag{3.2}$$

In the above equations, N denotes the total number of sentences in the generated answer. Each a_i represents the i-th sentence in the answer, while p_j refers to the j-th sentence in the source passage. $P_{\text{entailment}}(p_j, a_i)$ and $P_{\text{contradiction}}(p_j, a_i)$ are the probabilities predicted by the model, that sentence a_i is entailed by or contradicts sentence p_j, respectively. The $\max_j$ operator selects the maximum probability over all source sentences p_j for each answer sentence a_i.

In the second step, the Obligation Coverage Score (OC_s) is computed to assess completeness. An obligation classifier is used to label each sentence in the passage as either an obligation or non-obligation. This classifier is a LegalBERT model [Cha+20] fine-tuned on a synthetic dataset generated by prompting GPT-4 Turbo in a zero-shot setting. Each labeled obligation sentence is then compared with the answer sentences using the microsoft/deberta-large-mnli model [He+21]. If the maximum entailment probability for an obligation exceeds 0.7, it is considered covered. The OC_s is defined as the fraction of covered obligations out of all obligations present:

$$OC_s = \frac{1}{M} \sum_{k=1}^{M} \mathbb{1}\left(\max_{l} P_{\text{entailment}}(o_k, a_l) > 0.7 \right) \tag{3.3}$$

where M is the total number of obligation sentences in the source passage. $P_{\text{entailment}}(o_k, a_l)$ denotes the probability that the k-th obligation sentence in the source passage (o_k) is entailed by the l-th sentence in the answer (a_l). The $\max_l$ function selects the highest entailment probability for each obligation sentence across all sentences in the answer. The indicator function $\mathbb{1}$ returns 1 if the highest entailment score exceeds 0.7, meaning the obligation is considered covered.

Finally, the overall RePASs score integrates these three components in a normalized form:

$$\text{RePASs} = \frac{E_s - C_s + OC_s + 1}{3} \tag{3.4}$$

This equation rewards grounding and obligation coverage while penalizing contradictions, producing a single score between 0 and 1 that reflects the quality of regulatory answers in accuracy, consistency, and completeness.

4 Implemented methods and systems

This chapter describes the methods and systems implemented in this thesis. The methods presented in Sections 4.2.3 and 4.4 are based on AUEB NLP group's submissions to the retrieval and generation sub-tasks for RIRAG 2025 respectively, forming a complete RAG system. The group ranked 4th in the retrieval sub-task and 3rd in the generation sub-task among 19 participants. The chapter is divided into four sections that mirror the workflow and pipeline of a RAG system.

4.1 Data preparation

A common - if not necessary - initial step in RAG systems is the analysis and preparation of the retrieval corpus upon which the retriever will operate. In this work, the retrieval corpus consists of the 13,732 passages provided by the ObliQA dataset, which is described in Section 3.2. This section outlines how the dataset was pre-processed, enriched, and embedded to support semantic retrieval within the proposed system. The neural retrievers examined in this work include text-embeddings-3-large by Openai and voyage-law-2, voyage-finance-2 and voyage-3 by VoyageAI.[1][2]

To enrich the passages with additional features useful for chunking and retrieval, a series of preprocessing steps were applied. Firstly, context was added to the passage text by prepending it with its identifier (which can be found in the "PassageID" field; see Figure 3.1), resulting in a single combined string field. This approach ensures that questions explicitly referencing a passage by its identifier can be accurately grounded in the corresponding text. Next, sentence segmentation was performed using the SpaCy sentencizer, which helped to split the text into sentences.[3] Based on this segmentation, several useful statistics were computed: the total number of characters, the number of tokens (as counted by each model's respective tokenizer) and sentence count per passage. These metrics were later used for token-aware chunking.

[1] https://platform.openai.com/docs/models/text-embedding-3-large
[2] https://docs.voyageai.com/docs/embeddings
[3] https://spacy.io/api/sentencizer

Due to the maximum input length of each embedding model, a custom chunking algorithm was employed to split longer passages into smaller, model-compatible segments, ensuring that all resulting chunks retained the same identifier. The algorithm iteratively grouped sentences while keeping the total token count per chunk within the predefined limit. This method ensured that sentence boundaries were respected and that no sentence was split in the middle, preserving semantic coherence. However, due to the difference in context length between OpenAI's model and VoyageAI's models, the total passage chunks were different, which resulted in problems with rank-fusion during retrieval. These problems are described in Section 4.2.3. Each resulting chunk was also annotated with metadata, such as character count, word count, and token count, to support downstream inspection and filtering. This chunking strategy was adopted given the fact that information in legal and especially regulatory contexts should not be lost; as it is critical to the success of the system and the satisfaction of the user.

Following the chunking process, all sentence-based chunks were embedded using each of the four embedding models. The resulting embeddings were stored both as PyTorch tensors and as part of a structured pandas DataFrame. This DataFrame, containing both vector and textual metadata, was saved to disk as a .csv file for easy integration into the retrieval component of the system.

A similar embedding procedure was followed for the questions in the ObliQA dataset development and test splits. Each question was encoded using the same model but in query mode in the case of the Voyage models. This ensured that the vector representations of the questions were aligned with those of the passage chunks, allowing for effective similarity computation via the cosine similarity/dot product. The question embeddings were also stored in a separate DataFrame for use during parameter tuning, evaluation and testing.

4.2 Retrieval

4.2.1 BM25

To establish a strong baseline for passage retrieval, the BM25 ranking function was implemented using the rank_bm25 library.[4] BM25, described in Section 2.2.1, is a retrieval model based on term-frequency that assigns a relevance score between a query and a document based on the frequency with which query terms appear in the document, adjusted by document length. This method does not rely on learned embeddings and instead focuses on exact term matching, making it a competitive baseline against more complex neural retrievers. The use of multiple tokenizers can help investigate the impact of different tokenization methods on BM25 retrieval performance.

[4] `https://pypi.org/project/rank-bm25/`

Each sentence chunk in the retrieval corpus was tokenized using one of several tokenizers, depending on the experimental configuration. For the baseline, nltk.word_tokenize was applied to both passage chunks and the development and test questions. In other settings, tokenization was performed using OpenAI's tiktoken with the cl100k_base tokenizer, or the respective tokenizers of the voyage-finance-2, voyage-law-2, and voyage-3 models from the Hugging Face Transformers library.[5][6] These tokenizers correspond to the embedding models used in later stages. Each tokenizer influenced token-level features such as stop words, punctuation, and domain-specific terms, thus affecting retrieval behavior.

After tokenization, the tokenized sentence chunks were passed into the BM25Okapi model, which pre-computes term frequency statistics across the corpus. At query time, each question was tokenized in a similar way and scored against all available passage chunks. The top-scoring chunks were then sorted by score in descending order. Although multiple chunks might originate from the same passage, the system only retained the top scoring chunk per PassageID in the final results. This ensured completeness in the returned answers and better alignment with the passage-level evaluation metrics used later.

For each configuration, the top-k results ($k = 10$) were written to .trec files using the TREC format. The same procedure was applied to the test set for generalization assessment.

4.2.2 Neural Retrieval

In addition to lexical retrieval, neural retrievers based on dense representations were employed to measure semantic similarity between questions and passage chunks. Semantic similarity was calculated using the dot product, as the embeddings were L2-normalized. This approach operates on precomputed vector embeddings of passages and questions, as described in Section 4.1.

Each question in the development and test sets was embedded into a vector and then compared with the entire set of passage chunk embeddings using a dot product operation. This produced a similarity score for each passage chunk. Following the approach used in the BM25 experiments, the top-scoring passages were then sorted and filtered to include only the top-k most relevant passages per query, where $k = 10$.

This process was repeated across all embedding models to evaluate how different vector representations influence neural retrieval performance. Each retrieval run produced a ranked list of results written in a .trec file in the standard TREC format, making the results compatible with official evaluation tools such as trec_eval. As in the BM25 experiments, the same procedure was applied to the test set to assess generalization performance.

[5] https://github.com/openai/tiktoken
[6] https://huggingface.co/voyageai

4.2.3 Fusion-Based Retrieval

In the context of regulatory question-answering, combining lexical and semantic retrieval methods can significantly enhance performance by capturing different aspects of relevance. More specifically, lexical models such as BM25 are effective in matching exact terms and phrases, while neural retrievers utilize dense vector representations to capture broader meaning and paraphrastic variations. Hybrid retrieval systems that combine these two ways can exploit their strengths and mitigate individual weaknesses. In this study, hybrid retrieval was examined by pairing each dense retriever with a BM25 retriever tokenized using the same tokenizer employed for generating its dense embeddings. This alignment ensured consistency in input representation. In addition, other combinations of retrievers were explored to provide a complete analysis of how different retrieval strategies interact. Although special emphasis was placed on hybrid retrieval involving lexical and semantic signals, purely semantic combinations were also examined to assess their performance.

As already described in Section 2.2.3, Reciprocal Rank Fusion (RRF) is a rank-based fusion method that aggregates rankings from multiple retrieval systems by assigning higher importance to results that appear near the top of any individual ranked list. It has become a standard approach in retrieval systems due to its simplicity and robustness. In this work, RRF was used to combine retrieval outputs in several configurations.

First, hybrid retrieval using RRF was implemented by fusing each semantic retriever with its BM25 counterpart, i.e., BM25 models tokenized text using the same tokenizer as the dense embedding model. This hybrid setup allowed the system to combine exact term matching with semantic similarity upon the same tokenization basis. Next, all pairwise combinations of neural retrievers were tested using RRF, offering insight into how models with different domain specializations (e.g., legal, financial, general-purpose) could complement each other. To further explore the potential of rank fusion, triple combinations were tested. These included hybrid triples combining two neural retrievers and a single BM25 retriever with the best-performing BM25 variant (based on prior evaluations) and purely semantic triples formed by fusing three neural retrievers.

Unlike RRF, which operates on ranks, score-based rank fusion, which was described in Section 2.2.3, aggregates the raw similarity scores from different retrievers using weighted combinations. This method allows for more control over the influence of each retriever. Score-based fusion was applied to the same retrieval configurations examined under RRF, including BM25–dense hybrids, dense retriever pairs, and mixed combinations of hybrid and dense retriever triples. To handle three retrievers instead of two, the score-based fusion described by Equation 2.3 was expanded as follows:

$$s(d) = a \cdot \hat{s}_{\mathrm{x}}(d) + b \cdot \hat{s}_{\mathrm{y}}(d) + (1 - (a + b)) \cdot \hat{s}_{\mathrm{z}}(d), \tag{4.1}$$

where $a, b \in [0, 1]$ are fusion weights with the constraint $a + b \leq 1$, $\hat{s}_{\mathrm{x}}(d)$, $\hat{s}_{\mathrm{y}}(d)$ and $\hat{s}_{\mathrm{z}}(d)$ are the normalized scores produced by the retrievers for document d, and $s(d)$ is the final fused score assigned to document d.

However, It was not feasible to combine the OpenAI embedding model with the Voyage-based neural retrievers because they use different embedding dimensions and context lengths. These differences led to a mismatch in the number of passage chunks created during the data preparation phase. Aligning them would have required a projection or alignment step that falls outside the scope of this work. Therefore, hybrid and fused retrieval experiments involving OpenAI were only conducted using BM25 with OpenAI tokenization or within standalone semantic combinations.

In general, these fusion experiments highlight the effectiveness of hybrid retrieval systems, particularly in complex domains like regulatory QA. By combining domain-specific models such as voyage-law-2 and voyage-finance-2 with more general models like voyage-3 and OpenAI's text-embeddings-3-large, the system can achieve a more nuanced understanding of the content. This is especially important for datasets like ObliQA, which span both legal and financial domains, and thus benefit from models trained across different yet overlapping areas of expertise.

4.2.4 Re-ranking

In information retrieval pipelines, a two-stage architecture is often used. In such systems, the initial retrieval phase casts a wide net using fast lexical or semantic search techniques to gather potentially relevant candidates. These candidates are then passed on to a second-stage reranker, which performs a more fine-grained analysis to reorder and refine the results based on their actual relevance. This approach enables systems to benefit from the recall of broad retrieval methods while leveraging the precision of more computationally expensive models like cross-encoders.

To implement this architecture, the best performing retrieval method from those presented above was chosen. In the retrieval system the number of initially retrieved passages chunks was increased to 100, allowing a wider pool of candidates and increasing recall. From this expanded set, only the top-k unique PassageIDs were preserved.

After the first stage, the top retrieved passages were passed to a second-stage reranker to refine their ordering. This reranker was based on the voyage-rerank-2 model provided

by Voyage AI, a transformer-based cross-encoder.[7] Unlike bi-encoders, which embed queries and documents independently, cross-encoders process each query-document pair jointly through multiple attention layers. This architecture allows the model to capture fine-grained interactions between the question and the passage, typically yielding more accurate relevance estimates. In this setup, the reranker was applied to the retrieval results from the first stage. The reranker returned the final 10 passages per query, ensuring that the most relevant content was surfaced, while maximizing recall via the broader first-stage search.

The re-ranking pipeline was executed for various initial retrieval cut-offs ranging from 10 to 100 passage chunk candidates. The goal was to explore how expanding the initial retrieval depth impacts the quality of the re-ranked results.

This modular approach to retrieval demonstrates how combining lexical and neural retrievers through rank fusion methods, followed by a powerful reranker, can significantly enhance retrieval performance in complex domains such as legal and financial compliance. The ability to combine domain-specific models with broader models and to refine retrieval through cross-encoder-based re-ranking proved crucial for adapting to the mixed nature of the ObliQA dataset.

4.3 Preprocessing

4.3.1 Filtering and Obligation Extraction

Before passing the retrieved results to the answer generation module, a preprocessing step was applied to filter the candidate set of passages and focus on sentences that explicitly refer to obligations. This step was motivated by the nature of the ObliQA dataset, which is comprised of regulatory documents. Obligation clauses in these documents are central to answering compliance-related questions and essential for minimizing the risk of excluding them in generated responses.

The preprocessing step consists of two main components: passage filtering and obligation extraction. For the filtering step, the passages retrieved for each query were first ranked by their relevance scores. A filtering function was then applied to retain only those passages that (i) exceeded a fixed minimum relevance threshold and (ii) did not exhibit a sudden drop in relevance compared to the preceding passage chunk in the ranking. The maximum allowed drop in score was controlled by a hyperparameter. This strategy follows the approach described by Gokhan et al. [Gok+24] and helps to ensure that only top-quality passages are considered for downstream processing. If no passage remained after filtering,

[7] https://blog.voyageai.com/2024/09/30/rerank-2/

the highest-ranked one was kept to guarantee that at least one candidate was available In the second step, each passage that passed the filter was processed to extract sentences that contained explicit obligations. For this purpose, a LegalBERT-based obligation classifier was used, which had been fine-tuned on regulatory texts and is the same classifier that is used to evaluate RePASs (Section 3.3). The classifier labeled each sentence as either containing an obligation or not. If at least one obligation sentence was found, this was kept and stored as refined evidence set. If no obligations were detected in any of the retrieved passages, the original passage was preserved in full.

The output of the preprocessing stage is a structured list of dictionaries, one per question, each containing the question text, the top filtered retrieved passage chunks, and, optionally, the extracted obligation sentences. This setup provided a cleaner and more focused context for downstream modules, particularly beneficial in regulatory question-answering where precise legal obligations often form the basis of correct answers.

4.3.2 Prompt optimization

Prompt optimization is applied as the final step in the preprocessing pipeline to complete and evaluate the obligation extraction process, ensuring that the extracted obligations presented to the LLM can truly maximize the quality of the response. The system prompt (see Prompts) proposed in the original paper [Gok+24] was designed to be more general, enabling it to handle entire passages rather than focusing exclusively on obligations, which are central to answering compliance questions. Consequently, it became necessary to optimize the system prompt to better suit the new context provided to the LLM.

This optimization process was carried out through manual experimentation with GPT-4o. Initially, questions from the development set — already answered using the original prompt applied to the golden passages — were used as a starting point. GPT-4o was provided with the question, its corresponding answer, the golden passages, the values of the RePASs components as well as the composite score for that specific answer, and a short explanation about each component of the metric (Section 3.3). The optimization involved asking GPT-4o to suggest modifications or additions to the prompt that could improve each specific component of RePASs or the composite score.This process was iterative, using three different question-answer pairs and the necessary data for each cycle. With each improved prompt, the answers to these questions were regenerated and re-evaluated. The iteration continued until GPT-4o's suggestions became repetitive and no further improvements in the composite score were observed. Due to its length, an example without each question's information for an iteration of this process can be seen in Figure 4.1. Alternatively, libraries that provide prompt optimization, such as DSPy, could be used.[8]

[8] https://dspy.ai/

Your task is to help me improve the system prompt for the generation component of a RAG system that uses gpt-4o. You will be given three answers and the information given to generate them as well as their evaluation scores.

Generated answers are evaluated on RePASs (Regulatory Passage Answer Stability Score) that evaluates how well an answer aligns with regulatory source passages by combining three components:

Entailment Score:
For each answer sentence, the system finds the strongest supporting sentence in the source passage using an NLI model. This score reflects how well each answer sentence is justified by the source. A higher value means the answer is well-grounded in the passage.

Contradiction Score:
For each answer sentence, the system checks whether any passage sentence contradicts it. The strongest contradiction probability is kept per answer sentence. The average of these values shows how much the answer conflicts with the source. A lower value is better.

Obligation Coverage Score:
Regulatory passages often contain obligations. A LegalBERT-based classifier detects obligation sentences in the passage. Each obligation is checked against the answer using NLI. If the answer covers (entails) that obligation with sufficient confidence, it is considered covered. The score is the fraction of obligations from the passage that the answer correctly reflects.

Final RePASs Score:
The three components are combined into one stability score. Contradictions reduce the score, while entailment and obligation coverage increase it. The result is normalized to fall between 0 and 1, preferring answers that are accurate, contradiction-free, and complete.

Answer 1:
[Prompt] [Question and Retrieved Passages] [Answer] [Evaluation scores]

Answer 2:
[Prompt] [Question and Retrieved Passages] [Answer] [Evaluation scores]

Answer 3:
[Prompt] [Question and Retrieved Passages] [Answer] [Evaluation scores]

Tell me how to improve the prompt so that you/gpt-4o can understand better how to cover obligations more efficiently and increase the Obligation Coverage Score.

Fig. 4.1: Prompt structure for manual prompt optimization

4.4 Generation

4.4.1 Naive Obligation Concatenation

The Naive Obligation Concatenation (NOC) method is a simple adversarial approach to answer generation that directly leverages the output of the obligation extraction step described in Section 4.3.1 and exploits the fact that RePASs, as a reference-free metric, does not verify alignment with gold-standard passages. For each question, it constructs an answer by simply concatenating the sentences identified as obligations within the top retrieved passages. If no obligations are detected for a passage, the passage itself is used. This process generates one answer per question without any form of generative modeling or language inference, relying just on the extracted content.

This technique adopts the spirit of adversarial evaluation, as discussed in the related work chapter (Section 2.2.6). Rather than attacking the QA model directly, NOC is designed to challenge the evaluation metric RePASs, which emphasizes obligations as a core component of answer quality. Given that the answers are composed entirely of obligation sentences, the expected RePASs obligation sub-metric score should be almost perfect. Additionally, since obligations typically represent non-conflicting regulatory statements, it is anticipated that the generated responses would also yield a low contradiction score.

NOC uses the structure of the retrieval and evaluation steps to test how well RePASs can tell the difference between good, understandable and accurate answers, and the ones that just collect related facts. It should be stressed that NOC represents a method that is not intended for practical deployment but for testing the limits and sensitivities of the proposed scoring framework.

4.4.2 LLM Obligation Concatenation

The LLM Obligation Concatenation (LOC) algorithm builds upon the adversarial nature of NOC (Section 4.4.1), addressing its main limitation: the lack of contextualization and direct relevance to the posed question. While NOC merely aggregates obligation sentences, LOC introduces an LLM-based reasoning step that attempts to generate question-specific responses grounded in each extracted obligation. The motivation behind this method is to maintain the high obligation coverage scores expected under the RePASs metric, while at the same time improving the entailment score and making the answers more understandable by a non-expert.

For each extracted obligation associated with a question, the model prompts an LLM, which in this case is Gemini 2.5 Flash-Lite, to generate a partial answer using that obligation as

context and a specialized prompt (see Prompts).[9] If the generated answer does not satisfy the required obligation coverage, as measured by the same coverage-checking mechanism used in RePASs (see Equation 3.3), the model is re-prompted with the same input. In this setting, each partial answer corresponds to a single obligation and the coverage criterion reduces to checking that the answer entails the obligation with probability above 0.7. This retry process is repeated up to a predefined number of attempts, set to $K = 3$ in the current implementation. Once the partial answer passes the coverage check or the retry limit is reached, it is accepted and added to the list of partial responses. After processing all obligations, the final answer is formed by concatenating all verified partial answers.

Algorithm 1 LLM Obligation Concatenation (LOC)

Require: Preprocessed dataset with extracted obligations
Require: LLM model, max retries $K : integer$
for all item in preprocessed set **do**
    Initialize empty list answers_per_obligation
    for all obligation in item.Obligations **do**
        answer ← empty string
        coverage_flag ← False
        tries ← 0
        while not coverage_flag **and** tries $< K$ **do**
            prompt ← GeneratePrompt(question=item.Question, context=obligation)
            answer ← CallLLM(prompt)
            coverage_flag ← IsCovered(obligation, answer)
            tries ← tries + 1
        Append answer to answers_per_obligation
    Set item.PartialAnswers ← answers_per_obligation

4.4.3 Verify and Refine with RePASs

The Verify and Refine with RePASs (VRR) algorithm introduces an iterative answer generation process inspired by self-consistency and self-refinement methods proposed in recent work on large language models [Wan+23; Mad+24]. Unlike previous methods that either concatenate obligations or generate per-obligation responses, VRR formulates answer generation as a two-phase optimization process aimed at maximizing compliance with RePASs criteria. The two key phases are verification and refinement.

In the verification step, the model first generates N different answers using the full set of extracted obligations, the given question as context (Section 4.3.2) and the optimized prompt (see Prompt 6). These answers are independently evaluated based on the retrieved passages using RePASs as a proxy metric. Among the N candidate answers, the one achieving the highest composite RePASs score is selected for further refinement. This selection mechanism draws on the principles of self-consistency, where multiple reasoning paths are explored and the most reliable output is kept.

[9] https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-flash-lite

Once the best answer is identified, the refinement stage begins. First, the algorithm computes a sentence-level contradiction profile using natural language inference (NLI) models, comparing the answer against the original retrieved passages. Sentences exhibiting a contradiction score higher than the average or a predefined threshold are flagged for removal. In this work, the threshold is set as the average contradiction score computed across the dataset for the purposes of RIRAG 2025. However, in order to maintain compliance, the algorithm checks whether these sentences also fulfill any obligation. Otherwise, they are removed to ensure better factual alignment and to increase obligation coverage.

Next, the model addresses any remaining gaps in the coverage of obligations through the insertion of obligations. Using the same coverage function as in RePASs, the algorithm detects whether all obligations from the original passages are reflected in the current answer. If uncovered obligations are found, a prompt different from the one in the first phase is constructed in which the LLM is tasked with rewriting or expanding the current answer to include the missing information (see Prompts). This final step is inspired by iterative refinement techniques from recent instruction evolution frameworks [Xu+24] and ensures both faithfulness and completeness in regulatory responses. Experiments showed that adding obligations reduces the overall RePASs score. Therefore, in the final VRR iteration, this step is omitted and only contradiction refinement is applied.

Algorithm 2 Verify and Refine with RePASs (VRR)

Require: Preprocessed dataset with obligations and passages
Require: RePASs evaluation function, LLM generation functions
for all item in preprocessed **do**
    Generate N answers using all obligations as context
    for $i = 1$ to N **do**
        Prompt LLM with question and obligations
        Append generated answer to item.Answers
    Evaluate all answers using RePASs
    Select best answer based on highest score
    Set item.Answer to selected answer
for $i = 1$ to ref_iter **do**
    Compute dataset average contradiction score
    Set average contradiction score as the threshold
    for all item in preprocessed **do**
        Compare answer and retrieved passages using NLI
        Remove high-contradiction sentences if not covering obligations
        if $i < ref\_iter$ **then**
            Identify uncovered obligations
            Prompt LLM to revise answer with missing obligations
        Set item.Answer to updated version

The VRR method demonstrates particular effectiveness in domains like regulatory compliance, where ensuring high precision and accurate alignment with obligations is not only technically challenging but also crucial for downstream applications such as automated legal reasoning and risk mitigation.

5 Experiments

In this chapter, the experimental framework used to assess each stage of the proposed pipeline is presented. The evaluation focuses on three main components: retrieval, preprocessing, and generation. For each one, relevant metrics are applied and the results of various system configurations are analyzed. The results offer a comparative view of their performance and help highlight the strengths and limitations of methodological choices.

5.1 Datasets

All experiments in this thesis were performed using the ObliQA dataset [Gok+24], a benchmark specifically designed for evaluating question-answering systems in regulatory contexts, as discussed in Section 3.2. The development set was utilized for hyperparameter tuning and comparative validation of retrieval and generation variants. For the preprocessing and generation components, only the first 100 questions of the development set were used for hyperparameter tuning, the rest were discarded. After selecting the best-performing configurations, the test set was reserved for the final evaluation to ensure a fair assessment of the effectiveness of each method under standardized conditions.

5.2 Evaluation measures

For the evaluation of the methods developed for the retrieval component, Recall @ 10 and MAP @ 10 were used following the framework defined by RIRAG 2025 (Chapter 3). Both metrics are widely adopted in IR and question-answering tasks for quantifying the quality and completeness of retrieved results. For the preprocessing and generation steps, RePASs, which is thoroughly explained in Section 3.3, was used.

Recall@10 (R@10) measures the proportion of relevant documents that are successfully retrieved among the top-10 results. It reflects how well the system is able to identify the documents that matter most for each query. Formally, it is defined as:

$$\text{Recall@k} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \tag{5.1}$$

A score of 1 means that all relevant documents appear in the top-k, while a score of 0 means that none do. Although, a high recall can be achieved easily by retrieving many documents, using a fixed top-k cutoff like Recall@10 ensures a realistic and fair comparison.

Mean Average Precision at 10 (MAP@10) evaluates both whether relevant documents are retrieved and how high they appear in the ranking. For a single query, the average precision is calculated by averaging the precision values at each rank where a relevant document is found (up to the 10th result). MAP@10 is then the mean of the average percision scores across all queries:

$$\text{MAP@k} = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{k_j} \sum_{i=1}^{k_j} \text{Precision}(R_{ji}) \tag{5.2}$$

Here, $|Q|$ is the total number of queries, k_j is the number of relevant documents for query j, and $\text{Precision}(R_{ji})$ is the precision at the rank position where the i-th relevant document occurs for query j. Unlike recall, MAP rewards systems that retrieve relevant documents early in the ranking, making it a strong indicator of overall ranking quality.

5.3 Experimental results

5.3.1 Retrieval

BM25

The results of BM25-based lexical retrieval experiments, shown in Table 5.1, confirm the robustness of this classical lexical approach. In all the experiment the default values for k_1 and b were used (see Figure 2.1), without any specific tuning. Among all tokenizers tested, the variant using the voyage-law-2 tokenizer achieved the highest Recall@10 score in the test set, while the NLTK tokenizer achieved the highest MAP@10 score. This suggests that aligning the tokenizer with the domain improves retrieval effectiveness, likely due to better handling of domain-specific terminology. Although differences between tokenizers are small, the performance of voyage-law-2 demonstrates the importance of matching tokenization schemes to the vocabulary of the corpus and queries. In general, the BM25 model proved to be a competitive baseline, even against more complex semantic retrieval models, showcasing its long-standing usefulness in traditional information retrieval tasks.

Tokenizer	Recall@10	MAP@10
Nltk	0.699	**0.558**
tiktoken-cl100k_base	0.700	0.547
voyage-law-2	**0.707**	0.548
voyage-finance-2	0.701	0.545
voyage-3	0.697	0.543

Tab. 5.1: Retrieval performance of BM25 with different tokenizers on the test set.

Dense retrievers

The results in Table 5.2 highlight the effectiveness of neural retrieval methods using commercially available embedding models for semantic search in regulatory NLP. Achieving a Recall@10 score of 0.789, voyage-finance-2 ranked highest in recall, while voyage-3 achieved the best MAP@10 score. This supports the hypothesis that specialized commercial models trained on financial or legal corpora, such as voyage-finance-2 and voyage-law-2, are better suited for retrieving semantically relevant regulatory content, likely due to their alignment with domain-specific language. Compared to the strong baseline of text-embedding-3-large, which lags behind in both metrics, the results suggest that even famous general-purpose neural retrievers can be outperformed by commercially fine-tuned alternatives. Ultimately, these experiments confirm the practical utility of commercial neural retrievers in high-stakes, knowledge-intensive domains like law and finance.

Embedding model	Recall@10	MAP@10
text-embedding-3-large	0.738	0.573
voyage-law-2	0.770	0.627
voyage-finance-2	**0.789**	0.655
voyage-3	0.788	**0.656**

Tab. 5.2: Performance of single neural retrievers on the test set.

Fusion methods

Table 5.3 highlights that the combination of multiple retrievers using Reciprocal Rank Fusion (RRF) leads to strong improvements over individual models, confirming what has been discussed in the literature about hybrid search (Section 2.2.3). The best performing combinations were those that fused neural retrievers with complementary domain expertise, such as voyage-finance-2 and voyage-3, or even better, when combined with a BM25 retriever using the best tokenizer (voyage-law-2). For example, the triple combination of BM25 with voyage-finance-2 and voyage-3 reached the highest Recall@10 of 0.805 on the test set. This shows that even though neural retrievers perform well on their own, combining them, especially with a lexical model, adds robustness and helps retrieve a more complete set of relevant passages. These results clearly suggest that smart combinations using RRF can outperform even the best single embedding model.

Method	Recall@10	MAP@10
BM25-text-embedding-3-large	0.763	0.598
BM25-voyage-law-2	0.782	0.625
BM25-voyage-finance-2	0.782	0.631
BM25-voyage-3	0.776	0.629
text-embedding-3-large-voyage-law-2	0.779	0.619
text-embedding-3-large-voyage-finance-2	0.782	0.629
text-embedding-3-large-voyage-3	0.790	0.634
voyage-law-2-voyage-finance-2	0.790	0.654
voyage-law-2-voyage-3	0.795	0.659
voyage-finance-2-voyage-3	0.798	0.667
BM25-text-embedding-3-large-voyage-law-2	0.797	0.638
BM25-text-embedding-3-large-voyage-finance-2	0.801	0.645
BM25-text-embedding-3-large-voyage-3	0.802	0.648
BM25-voyage-law-2-voyage-finance-2	0.801	0.653
BM25-voyage-law-2-voyage-3	0.801	0.656
BM25-voyage-finance-2-voyage-3	**0.805**	0.659
text-embedding-3-large-voyage-law-2-voyage-finance-2	0.789	0.643
text-embedding-3-large-voyage-law-2-voyage-3	0.794	0.648
text-embedding-3-large-voyage-finance-2-voyage-3	0.796	0.654
voyage-law-2-voyage-finance-2-voyage-3	0.798	**0.664**

Tab. 5.3: Reciprocal Rank Fusion RRF) results across lexical and neural retrievers on the test set.

The results shown in Tables 5.4 and 5.5 are based on the best-performing configurations for each rank-fusion method, identified through extensive hyperparameter tuning. A grid search was carried out over the interpolation weights a and b (Equation 4.1), with values ranging from 0.05 to 0.95 in increments of 0.05, using the development set. This process ensured that each retrieval configuration was evaluated under its most effective weighting scheme. Figure 5.1 provides a representative example of this tuning procedure for the BM25–voyage-finance-2 combination, where Recall@10 peaked at 0.807 for $a = 0.15$. The same tuning strategy was applied uniformly across all tested combinations.

Once the optimal interpolation weights were determined, they were used to assess generalization performance on the test set. Score-based rank fusion consistently led to performance gains, especially when integrating BM25 with neural retrievers that utilized the same tokenizer, as evidenced by the results in Tables 5.4 and 5.5. Among pairwise setups, BM25–voyage-finance-2 achieved the highest Recall@10 of 0.800, highlighting the strength of lexical and semantic complementarity. Notably, purely semantic combinations such as voyage-finance-2–voyage-3 also performed competitively, suggesting that model specialization plays a key role even without lexical signals. The best overall performance was achieved by the triple combination BM25–voyage-finance-2–voyage-3, reaching a Recall@10 of 0.808 and MAP@10 of 0.684 on the test set. These findings reinforce the idea that well-calibrated hybrid systems are highly beneficial for complex domains like

regulatory QA, where complementary lexical and semantic signals in the legal and financial domains, contribute to improved performance.

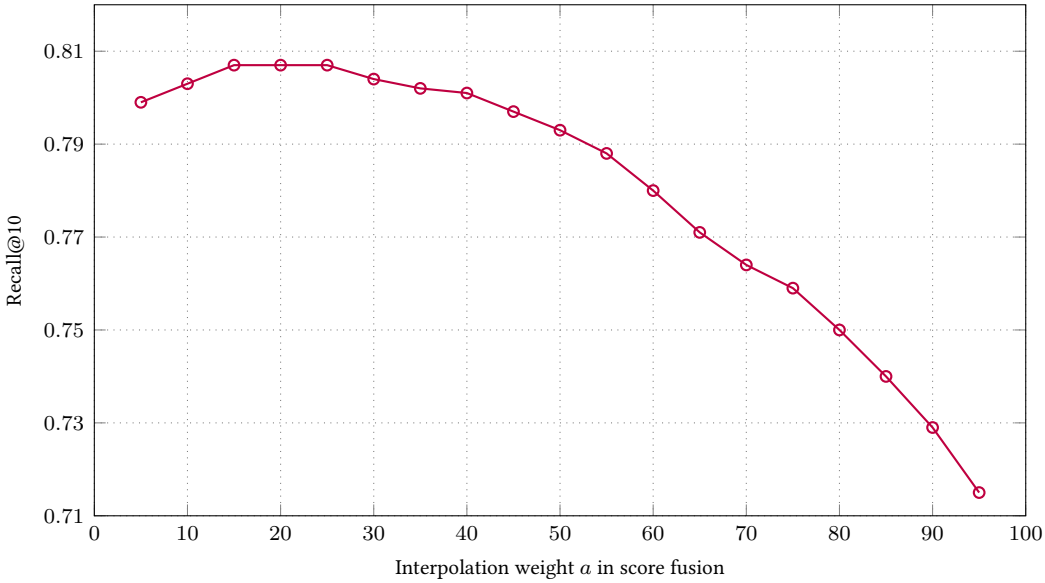

Fig. 5.1: Performance of BM25–voyage-finance-2 under varying interpolation weight a on the development set.

Method	a	Recall@10	MAP@10
BM25 - text-embedding-3-large	0.25	0.783	0.637
BM25 - voyage-3	0.20	0.797	0.676
BM25 - voyage-finance-2	0.15	**0.800**	**0.676**
BM25 - voyage-law-2	0.35	0.797	0.665
voyage-finance-2 - voyage-3	0.45	0.800	0.671
voyage-law-2 - voyage-3	0.50	0.797	0.665
voyage-law-2 - voyage-finance-2	0.30	0.790	0.660

Tab. 5.4: Performance of rank fusion combinations of lexical and neural retrievers on the test set.

Method	a	b	Recall@10	MAP@10
BM25 - voyage-law-2 - voyage-3	0.25	0.30	0.805	0.680
BM25 - voyage-law-2 - voyage-finance-2	0.20	0.25	0.805	0.681
BM25 - voyage-finance-2 - voyage-3	0.15	0.50	**0.808**	**0.684**
voyage-law-2 - voyage-finance-2 - voyage-3	0.10	0.40	0.800	0.671

Tab. 5.5: Triple rank fusion performance combining lexical and neural retrievers on the test set.

The inclusion of BM25 in both fusion strategies generally resulted in improved performance across both evaluation metrics. However, it should be noted that, in certain instances involving RRF in Table 5.3, the interpolation of ranks led to a decline in performance when BM25 was combined with individual neural retrievers. In contrast, the score-based rank fusion method, which relies on weighted interpolation of retrieval scores, did not have this issue, consistently yielding more stable and effective results. Among all evaluated combinations, the fusion of BM25, voyage-finance-2, and voyage-3 emerged as the most effective configuration under both fusion methods.

Re-ranking

The re-ranking experiment constitutes an additional stage of hyperparameter tuning, this time focusing on the retrieval depth parameter N, which determines how many top-ranked passages from the first-stage retriever are forwarded to the reranker. As shown in Figure 5.2, various values of N ranging from 10 to 100 were evaluated on the development set using the previously best-performing retrieval method: rank fusion of BM25, voyage-finance-2, and voyage-3. The reranker, voyage-rerank-2, was applied on the top-NN passages for each query, and the final re-ranked results were assessed using Recall@10. The highest performance was observed at N=20, achieving a Recall@10 of 0.818, slightly outperforming deeper candidate sets. As the initial retrieval pool expands, recall initially increases, peaking at 0.818 for a cut-off of 20 before gradually declining. This pattern suggests that while increasing the candidate pool allows the reranker to access more potentially relevant passages, beyond a certain point the benefits taper off due to the introduction of lower-quality candidates that introduce noise into the ranking.

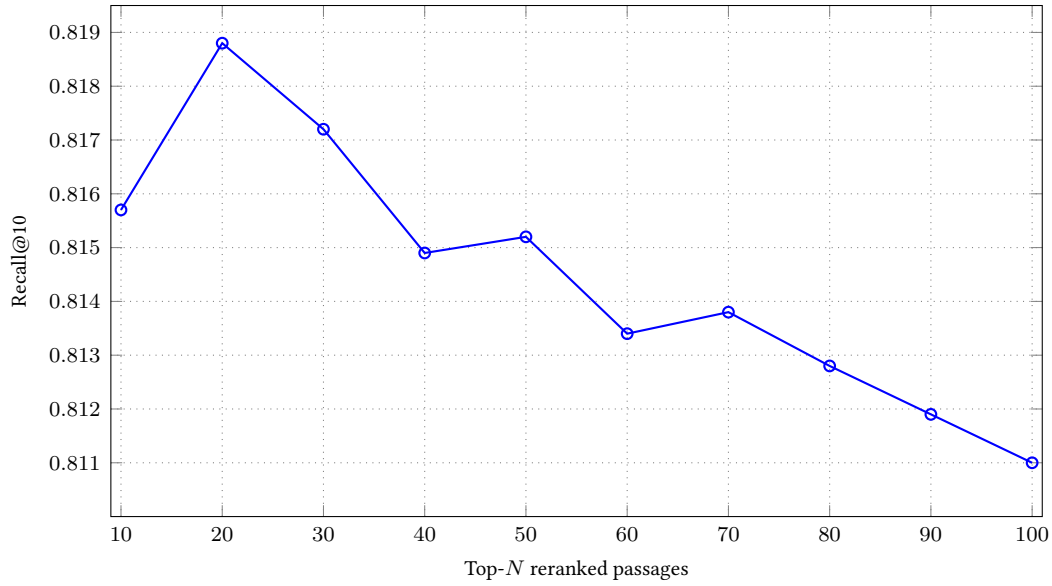

Fig. 5.2: Recall@10 for varying top-N passage cutoffs during re-ranking using voyage-rerank-2 on the development set.

Based on this development-set tuning, N=20 was selected as the optimal cutoff and subsequently applied to the test set. The resulting system achieved a Recall@10 of 0.813 and a MAP@10 of 0.710. This marks a clear improvement over the same set-up without re-ranking, which previously reached Recall@10 = 0.808 and MAP@10 = 0.684. These gains illustrate the effectiveness of cross-encoder re-ranking in refining document rankings after initial retrieval, validating its role as a critical component in two-stage architectures for regulatory QA. The observed improvements also reinforce the benefits of modular pipeline design, where even small adjustments to retrieval depth or ranking strategy can yield meaningful performance boosts in complex domain-specific tasks.

5.3.2 Preprocessing

The first step of the preprocessing pipeline involved filtering the set of retrieved passages based on their relevance scores, following a strategy inspired by Gokhan et al. [Gok+24]. Each candidate passage chunk was retained only if its relevance score surpassed a fixed threshold and did not exhibit a steep drop compared to the previous passage, with both parameters being subject to hyperparameter tuning. As shown in Table 5.6, the optimal configuration was found to be a threshold of 0.80 and a maximum drop of 0.1, which achieved the highest RePASs score of 0.512. It was also observed that larger maximum drop values tended to allow less relevant and potentially contradictory passages into the set, as evidenced by the increase in contradiction scores when max-drop increased. This supports the view that high-quality, precise passages are key in regulatory QA.

Threshold	Max-Drop	RePASs	Obligations	Entailment	Contradiction
0.5	0.1	0.487	0.194	0.477	0.207
0.5	0.2	0.470	0.204	**0.529**	0.323
0.5	0.3	0.459	0.228	0.509	0.357
0.6	0.1	0.482	0.179	0.463	0.195
0.6	0.2	0.480	0.207	0.493	0.258
0.6	0.3	0.474	0.219	0.496	0.292
0.7	0.1	0.502	0.216	0.475	0.185
0.7	0.2	0.468	0.199	0.441	0.234
0.7	0.3	0.481	0.198	0.486	0.242
0.8	0.1	**0.512**	0.237	0.453	0.154
0.8	0.2	0.503	**0.247**	0.450	0.187
0.8	0.3	0.489	0.205	0.459	0.196
0.9	0.1	0.497	0.219	0.398	0.126
0.9	0.2	0.502	0.222	0.396	0.111
0.9	0.3	0.507	0.230	0.401	**0.109**

Tab. 5.6: Performance of various threshold and max-drop configurations across different scoring dimensions for the development set.

In the second step, each filtered passage was further refined through the extraction of obligation-bearing sentences using a LegalBERT-based classifier trained on regulatory texts. This step aimed to reduce input noise by narrowing the focus of the context to legally meaningful content. To ensure compatibility with this refined input, the original prompt used in the baseline system (see Prompts) was re-optimized through iterative manual experimentation with GPT-4o, the resulting prompt can be seen in Figure 5.3. As reported in Table 5.7, three prompting strategies were compared: (i) the original prompt with unfiltered passages, (ii) the original prompt with extracted obligation sentences, and (iii) the optimized prompt with extracted obligations. Using the new prompt with extracted obligations significantly improved performance across most evaluation dimensions. Specifically, the entailment score increased from 0.611 to 0.723 and the overall RePASs score rose from 0.468 to 0.483, confirming that combining focused evidence with a task-adapted prompt yields more accurate and relevant answers. Interestingly, while obligation coverage decreased

slightly, the gain in semantic alignment (expressed by the entailment score) outweighed the trade-off, making this setting the configuration with the best performance.

Prompt for Obligation-Based Answer Generation

You are a regulatory compliance assistant. Your task is to provide a brief but concise and detailed answer to the Question, ensuring that all Obligations are fully addressed. Directly integrate each obligation into the response, ensuring no obligation is missed or implied. Avoid adding information beyond what is explicitly stated in the Obligations, and cite specific rules when necessary. Use the exact terminology and structure from the obligations where applicable, to ensure high alignment and logical consistency. Focus solely on the provided obligations to craft a response that is well-structured, concise, and free of contradictions.

Fig. 5.3: Obligation-focused prompt for regulatory QA.

Method	RePASs	Obligations	Entailment	Contradiction
Original prompt + passages	0.429	**0.189**	0.516	0.417
Original prompt + obligations	0.468	0.161	0.611	**0.367**
New prompt + obligations	**0.483**	0.177	**0.723**	0.451

Tab. 5.7: Comparison of prompting strategies for regulatory question answering. "+ passages" refers to providing the model with entire retrieved passages, while "+ obligations" refers to providing only obligation sentences extracted by the LegalBERT classifier.

In general, the pre-processing pipeline consisting of optimized passage filtering and focused obligation extraction, enhanced by a customized prompting strategy, led to measurable improvements in retrieval-based answer quality. These two steps acted synergistically to reduce irrelevant content and amplify signal strength around regulatory obligations. Moving forward, all answer generation experiments build upon this optimized preprocessing configuration, leveraging the best-performing filtering setup (threshold = 0.80, max-drop = 0.1) and the newly optimized prompt applied on extracted obligations. This configuration not only improves coverage of relevant obligations but also enhances the semantic alignment between questions and retrieved content. As a result, it serves as a strong foundation for downstream evaluations and ensures consistency across experimental conditions.

5.3.3 Generation

Naive Obligation Concatenation

The Naive Obligation Concatenation (NOC) method serves as an adversarial probe into the robustness of the RePASs evaluation metric, exploiting its reference-free nature to construct superficially compliant answers. As shown in Table 5.8, NOC achieves an almost perfect RePASs score of 0.939, substantially outperforming both the GPT-4o baseline (0.470) and even human experts (0.859). This superhuman performance extends across all sub-metrics:

Method	RePASs	Obligations	Entailment	Contradiction
GPT-4o baseline*	0.470	0.222	0.320	0.131
Human experts*	0.859	**1.000**	0.837	0.260
NOC	**0.939**	0.964	**0.985**	0.131
VRR	0.799	0.658	0.833	**0.093**
LOC	0.613	0.424	0.666	0.250

Tab. 5.8: Comparison of different generation methods using the RePASs metric on the test set. *Scores taken from Gokhan et al. [Gok+24].

the obligation coverage score nears the upper bound at 0.964, contradiction is minimized to 0.131, and entailment reaches an unexpected high of 0.985 despite the method's lack of generative reasoning. These results highlight a key limitation of RePASs: while designed to reward answers that align closely with regulatory obligations, it can be easily misled by methods that simply aggregate these obligations without providing coherent and human-readable answers, as such obligations commonly rely on domain-specific legal and financial language. The fact that NOC surpasses human performance without any understanding or curation confirms that RePASs, although useful for obligation-focused QA, is vulnerable to manipulation, mistaking surface-level lexical alignment for actual answer quality.

LLM Obligation Concatenation

The LLM Obligation Concatenation (LOC) method was developed to test whether grounding answers in obligation sentences while still allowing the LLM to reason improves generation quality. Unlike NOC, which simply concatenates obligation sentences, LOC leverages the LLM to reformulate and integrate these obligations into a coherent response conditioned on the question. It is important to recognize that NOC functions as an adversarial upper bound; by presenting every obligation at once, it artificially maximizes entailment in a way that is not practical for real-world question-answering, since it may not always be clear how an obligation sentence can help answer a question. The obligations in LOC's answer are no longer presented as disjoint sentences, but are contextualized and phrased in a way that aligns better with an answer format. As shown in Table 5.8, LOC achieves a RePASs score of 0.613, marking an improvement of +0.143 over the GPT-4o baseline (0.470). This improvement is primarily attributed to a substantial increase in obligation coverage (+0.202), suggesting that grounding responses in obligation content improves regulatory alignment. While this comes with an increase in contradiction (+0.121), the overall alignment still improves, as reflected in the entailment score that rises by +0.346, which is more than double from the baseline. These results suggest that, while LOC does not match the artificially strong performance of NOC, it provides adherence to regulatory content and thus significant improvement over the baseline.

Verify and Refine with RePASs

The Verify and Refine with RePASs (VRR) method demonstrates substantial improvements across all evaluation dimensions, as shown in Table 5.8. Compared to the GPT-4o baseline, VRR increases the RePASs score by +0.329 (from 0.470 to 0.799), obligation coverage by +0.436, entailment by +0.513, and reduces contradiction by -0.038. Unlike methods such as NOC or LOC, which rely on direct extraction or per-obligation prompting, VRR introduces an iterative, metric-driven generation process. As shown in Table 5.9, the initial verification step, which selects the best out of $N = 5$ generated answers, yields the greatest performance gain (+0.104 in RePASs), highlighting the effectiveness of candidate selection based on self-consistency principles and guided by RePASs scores.

Further improvements are achieved through the refinement phase, where contradiction removal reduces factual inconsistency by 0.079 (from 0.172 to 0.093) while preserving high entailment values above 0.83. Simultaneously, obligation-focused refinement raises obligation coverage to 0.658, a substantial +0.436 increase over the baseline and +0.154 over the verification step in the iteration with the highest RePASs. Each refinement stage, alternating between contradiction and obligation adjustments, contributes incrementally, with diminishing returns becoming evident by the fourth iteration, after which the process was stopped and the version with the highest RePASs is picked, as shown in the final rows of Table 5.9. In general, VRR ranks as the generative system with the highest performance, excluding LOC, which adversarially attacks RePASs, validating the importance of structured iterative refinement to achieve both completeness and factual precision in regulatory QA.

Step	RePASs	Obligations	Entailment	Contradiction
Preprocessing	0.624	0.375	0.670	0.172
Verify	0.728	0.504	0.802	0.119
Refine Contradictions 1	0.753	0.508	0.825	**0.072**
Refine Obligations 1	0.770	0.596	0.826	0.112
Refine Contradictions 2	0.779	0.600	0.826	0.087
Refine Obligations 2	0.770	0.596	0.826	0.112
Refine Contradictions 3	0.798	0.655	0.832	0.093
Refine Obligations 3	0.792	0.657	0.833	0.113
Refine Contradictions 4	**0.799**	0.658	**0.833**	0.093
Refine Obligations 4	0.797	**0.677**	0.831	0.116

Tab. 5.9: RePASs progression across VRR steps on the test set.

5.3.4 RIRAG 2025 Results and Comparison

For the evaluation of participants in RIRAG 2025 [Gok+25], the organizers released an evaluation set comprising 446 previously unseen questions, notably without providing their corresponding golden passages. The reference-free design of the RePASs metric enables a meaningful comparison between the results achieved in RIRAG 2025 and those of

System / Group	RePASs	Obligations	Entailment	Contradiction
GPT-4o baseline*	0.583	0.220	0.769	0.238
Human experts*	0.859	**1.000**	0.837	0.260
Indic aiDias	**0.973**	0.993	**0.987**	0.062
Ocean's Eleven	0.971	0.991	0.986	0.065
AUEB NLP - NOC	0.947	0.951	0.986	0.096
Thesis - VRR	0.807	0.688	0.827	0.093
AUEB NLP - VRR	0.639	0.502	0.446	**0.031**
AICOE	0.601	0.230	0.827	0.254
AUEB NLP - LOC	0.562	0.423	0.375	0.110

Tab. 5.10: Subtask 2 leaderboard on the evaluation set. *Baseline and expert scores from Gokhan et al. [Gok+24].

the system presented in this study, which establishes a new benchmark for non-adversarial methods. However, since the golden passages remain unavailable, a direct comparison between the retrieval component of the newly best-performing system and that of our submission is unfortunately not feasible.

Two submissions surpassed AUEB's baseline system on the RePASs leaderboard as seen in Table 5.10, both achieving near-perfect scores. Indic aiDias achieved the highest overall score (0.973), outperforming human experts in entailment while maintaining a low contradiction rate. Ocean's Eleven followed closely (0.971), with comparable scores across all metrics. Both systems appear to adopt similar strategies based on passage concatenation, optimizing surface-level alignment with gold answers. This approach likely contributes to their exceptional RePASs performance. However, these groups neither propose a concrete solution nor introduce a novel framework to enhance Repass; rather, they simply concatenate existing passages.

Step	Submission	Thesis
Preprocessing	0.506	0.626
Verify	0.611	0.734
Refine Contradictions 1	0.638	0.753
Refine Obligations 1	0.634	0.772
Refine Contradictions 2	0.643	0.782
Refine Obligations 2	0.637	0.788
Refine Contradictions 3	0.643	0.797
Refine Obligations 3	0.642	0.797
Refine Contradictions 4	0.647	0.807
Refine Obligations 4	0.641	0.801

Tab. 5.11: Comparison between the system submitted to RIRAG 2025 and the system developed in this thesis based on RePASs scores on the evaluation set.

This VRR method is further refined in the thesis system, which significantly improves on the original submission. A key difference lies in the use of Gemini 2.5 Flash-Lite as the generation model instead of GPT-4o. This change leads to more stable and accurate

completions when integrated with the verification and refinement stages. The performance advantage is evident from the preprocessing stage, where the thesis system begins with higher RePASs score of 0.626. This stronger initial baseline enables the refinement pipeline to produce more consistent improvements at each step, ultimately reaching a new peak performance of 0.807 (Table 5.11). These findings demonstrate that the change in generation model was instrumental in boosting performance and that VRR is a realistic approach that achieves the strongest results among all evaluated methods.

6 Conclusions

This thesis explored the field of Regulatory NLP, focusing on the development and evaluation of Retrieval-Augmented Generation systems for question-answering in compliance-sensitive domains. A central aim was to assess how effectively such systems can be adapted or built for highly specialized regulatory contexts, where precision is critical. Through systematic experimentation within the framework of the RIRAG-2025 shared task, the study demonstrated that specialized neural retrievers, particularly those fine-tuned on financial or legal corpora, can significantly outperform general-purpose models. Furthermore, the integration of dense and lexical models via rank fusion methods such as Reciprocal Rank Fusion and score-based interpolation was shown to produce competitive retrieval results. The addition of re-ranking modules further refined top-ranked results, validating the efficacy of two-stage retrieval architectures in regulatory domains.

Regarding answer generation, the thesis introduced and evaluated a variety of answer synthesis strategies, culminating in the Verify and Refine with RePASs (VRR) method. This iterative pipeline successfully improved grounding, entailment, and obligation coverage, while minimizing contradiction. Importantly, the thesis also provided a critical examination of the RePASs evaluation metric, showing that although it is well-suited for reference-free assessment of answers containing obligations, it is vulnerable to superficial optimization strategies such as the Naive Obligation Concatenation (NOC) method. In general, the findings highlight the importance of modular, domain-adaptive system design and robust evaluation practices for building reliable, high-precision QA systems in Regulatory NLP.

Future work

Future work could focus on improving evaluation by addressing the limitations of RePASs, which remains vulnerable to superficial response strategies. Developing more robust, semantically aware metrics would improve system reliability, particularly in high-stakes compliance scenarios. Another direction is creating similar benchmarks for non-English regulatory corpora, with Greek serving as a compelling example. Progress in this area would also benefit from the development of localized legal retrieval systems, language models, and obligation extraction tools tailored to specific jurisdictions.

Bibliography

[BCB94] Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. "Automatic combination of multiple ranked retrieval systems". In: *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Berlin, Heidelberg, 1994, pp. 173–181.

[Ber+13] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. "Semantic Parsing on Freebase from Question-Answer Pairs". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA, 2013, pp. 1533–1544.

[BGI23] Sebastian Bruch, Siyu Gai, and Amir Ingber. "An Analysis of Fusion Functions for Hybrid Retrieval". In: *ACM Trans. Inf. Syst.* 42.1 (Aug. 2023).

[BL04] Steven Bird and Edward Loper. "NLTK: The Natural Language Toolkit". In: *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. Barcelona, Spain, 2004, pp. 214–217.

[CCB09] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. "Reciprocal rank fusion outperforms condorcet and individual rank learning methods". In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA, 2009, pp. 758–759.

[Cha+18] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. *Adversarial Attacks and Defences: A Survey*. 2018. arXiv: 1810.00069 [cs.LG].

[Cha+20] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. "LEGAL-BERT: The Muppets straight out of Law School". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online, 2020, pp. 2898–2904.

[Gao+24] Yunfan Gao, Yun Xiong, Xinyu Gao, et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2024. arXiv: 2312.10997 [cs.CL].

[Goa+23] Catalina Goanta, Nikolaos Aletras, Ilias Chalkidis, Sofia Ranchordás, and Gerasimos Spanakis. "Regulation and NLP (RegNLP): Taming Large Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Singapore, 2023, pp. 8712–8724.

[Gok+24] Tuba Gokhan, Kexin Wang, Iryna Gurevych, and Ted Briscoe. *RegNLP in Action: Facilitating Compliance Through Automated Information Retrieval and Answer Generation*. 2024. arXiv: 2409.05677 [cs.CL].

[Gok+25] Tuba Gokhan, Kexin Wang, Iryna Gurevych, and Ted Briscoe. "Shared Task RIRAG-2025: Regulatory Information Retrieval and Answer Generation". In: *Proceedings of the 1st Regulatory NLP Workshop (RegNLP 2025)*. Abu Dhabi, UAE, 2025, pp. 1–4.

[HB23] Yichen Huang and Timothy Baldwin. "Robustness Tests for Automatic Machine Translation Metrics with Adversarial Attacks". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Singapore, 2023, pp. 5126–5135.

[He+21] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. "{DEBERTA}: {DECODING}-{ENHANCED} {BERT} {WITH} {DISENTANGLED} {ATTENTION}". In: *International Conference on Learning Representations*. 2021.

[HGC23] Pengcheng He, Jianfeng Gao, and Weizhu Chen. *DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing*. 2023. arXiv: 2111.09543 [cs.CL].

[Kal23] Joseph Kalmenovitz. "Regulatory Intensity and Firm-Specific Exposure". In: *The Review of Financial Studies* 36.8 (Jan. 2023), pp. 3311–3347. eprint: https://academic.oup.com/rfs/article-pdf/36/8/3311/50908023/hhad001.pdf.

[Kar+20] Vladimir Karpukhin, Barlas Oguz, Sewon Min, et al. "Dense Passage Retrieval for Open-Domain Question Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online, 2020, pp. 6769–6781.

[Kwi+19] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, et al. "Natural Questions: A Benchmark for Question Answering Research". In: (2019), pp. 452–466.

[LCT19] 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*. Florence, Italy, 2019, pp. 6086–6096.

[Lew+20] Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. "Retrieval-augmented generation for knowledge-intensive NLP tasks". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA, 2020, pp. 9459–9474.

[Li+20] Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. "BERT-ATTACK: Adversarial Attack Against BERT Using BERT". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online, 2020, pp. 6193–6202.

[LNY21] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. *Pretrained Transformers for Text Ranking: BERT and Beyond*. 2021. arXiv: 2010.06467 [cs.IR].

[Mad+24] Aman Madaan, Niket Tandon, Prakhar Gupta, et al. "SELF-REFINE: iterative refinement with self-feedback". In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA, 2024, pp. 46534–46594.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[NC20] Rodrigo Nogueira and Kyunghyun Cho. *Passage Re-ranking with BERT*. 2020. arXiv: 1901.04085 [cs.IR].

[Qua+24] Xin Quan, Marco Valentino, Louise A. Dennis, and Andre Freitas. "Verification and Refinement of Natural Language Explanations through LLM-Symbolic Theorem Proving". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA, 2024, pp. 2933–2958.

[RG19] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China, 2019, pp. 3982–3992.

[Rob+95] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. "Okapi at TREC-3". In: *Overview of the Third Text REtrieval Conference (TREC-3)*. Gaithersburg, Maryland, USA, 1995, pp. 109–126.

[Wan+23] Xuezhi Wang, Jason Wei, Dale Schuurmans, et al. "Self-Consistency Improves Chain of Thought Reasoning in Language Models". In: *The Eleventh International Conference on Learning Representations*. 2023.

[Wan+24] Ante Wang, Linfeng Song, Ye Tian, et al. "Self-Consistency Boosts Calibration for Math Reasoning". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Miami, Florida, USA, 2024, pp. 6023–6029.

[WZZ21] Shuai Wang, Shengyao Zhuang, and Guido Zuccon. "BERT-based Dense Retrievers Require Interpolation with BM25 for Effective Passage Retrieval". In: *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. New York, NY, USA, 2021, pp. 317–324.

[Xu+24] Can Xu, Qingfeng Sun, Kai Zheng, et al. "WizardLM: Empowering Large Pre-Trained Language Models to Follow Complex Instructions". In: *The Twelfth International Conference on Learning Representations*. 2024.

Prompts

Baseline prompt [Gok+24]

You are a regulatory compliance assistant. Provide a detailed answer for the question that fully integrates all the obligations and best practices from the given passages. Ensure your response is cohesive and directly addresses the question. Synthesize the information from all passages into a single, unified answer.

Prompt for obligations in the context (VRR)

You are a regulatory compliance assistant. Your task is to provide a brief but concise and detailed answer to the Question, ensuring that all Obligations are fully addressed. Directly integrate each obligation into the response, ensuring no obligation is missed or implied. Avoid adding information beyond what is explicitly stated in the Obligations, and cite specific rules when necessary. Use the exact terminology and structure from the obligations where applicable, to ensure high alignment and logical consistency. Focus solely on the provided obligations to craft a response that is well-structured, concise, and free of contradictions.

Prompt for inserting obligations (VRR)

You are a regulatory compliance assistant. Your task is to integrate the following Obligations that are missing from the Answer. You may change sentences or add new ones to cover all Obligations. Avoid adding changes or sentences that contradict the Answer and/or the Obligations.

Prompt that rewrites an obligation (LOC)

You are a regulatory compliance assistant. Your task is to construct a brief but concise response that addresses the Question by focusing exclusively on the specified Obligation. Ensure your response clearly identifies and explains the obligation, including any relevant conditions or restrictions. Avoid addressing unrelated aspects of the Question, and limit your response strictly to what is explicitly stated in the provided passage.

List of Acronyms

ADGM	Abu Dhabi Global Markets
AI	Artificial Intelligence
AQuA	Annotation Quality Assessment
AUEB	Athens University of Economics and Business
BM25	Best Matching 25
BLEURT	Bilingual Evaluation Understudy with Representations from Transformers
COLING	Conference on Computational Linguistics
COMET	Crosslingual Optimized Metric for Evaluation of Translation
CSV	Comma-Separated Values
DPR	Dense Passage Retrieval
GPT	Generative Pretrained Transformer
GSM8K	Grade School Math 8K
IDF	Inverse Document Frequency
IR	Information Retrieval
LOC	LLM Obligation Concatenation
LLM	Large Language Model

MAP Mean Average Precision

MRR Mean Reciprocal Rank

MT Machine Translation

NLI Natural Language Inference

NLP Natural Language Processing

NLTK Natural Language Toolkit

NOC Naive Obligation Concatenation

ObliQA Obligation-based Question-Answering

ORQA Operating Room Question Answering)

QA Question Answering

RAG Retrieval-Augmented Generation

RegNLP Regulatory Natural Language Processing

RePASs Regulatory Passage Answer Stability Score

RIRAG Regulatory Information Retrieval and Answer Generation

RRF Reciprocal Rank Fusion

SBERT Sentence-BERT

SVAMP Simple Variations on Arithmetic Math word Problems

TREC Text REtrieval Conference

VRR Verify and Refine with RePASs

List of Figures

List of Tables

List of Algorithms