Ablating Retrieval Modules for Temporal Conflict Resolution in Legal RAG

Dmytro Asieiev[1], Emilija Bareikaitė[1], Emilija Bareikaitė[1], Jadrine Kaburu[1], Jonas Urnėžius[1], and Aušra Šubonienė[1]

AAI Labs http://www.aai-labs.com

Abstract. Large language models (LLMs) are increasingly applied in domains such as legal, medical, and financial systems, yet they are prone to hallucinations, producing fluent but factually incorrect statements. This issue is particularly problematic in legal contexts, where strict requirements for factual accuracy and reliable citations exist. Retrieval-augmented generation (RAG) has been proposed as a solution to mitigate hallucinations by grounding model outputs in external documents; however, factual errors still occur when retrieved sources are conflicting, outdated, or irrelevant.

This paper investigates how different retrieval strategies influence factual correctness in legal question answering systems based on RAG. Ten variants of a legal RAG pipeline were evaluated using an ablation study on a dataset of 200 EurLex queries. The experiments compare dense and hybrid retrieval, metadata filtering, query expansion, and re-ranking techniques, and assess their impact using metrics such as Recall@5, non-answer rate, citation consistency, outdated retrieval rate, and cost per query.

The results show that hybrid retrieval substantially improves recall compared to dense retrieval, while metadata filtering reduces outdated results but can significantly lower recall when applied aggressively. Query expansion and re-ranking provide only limited improvements when retrieval quality is poor. Overall, the findings demonstrate that retrieval quality is the primary factor influencing the performance of legal RAG systems, highlighting the importance of optimizing retrieval strategies rather than relying solely on complex generation or reasoning mechanisms.

1 Introduction

In recent years, large language models (LLMs) have gained increasing attention for applications in various spheres, for instance, in enterprise, legal, medical and financial systems [1]. However, LLMs hallucinate, meaning they produce fluent, but factually incorrect text, which is especially alarming in legal systems which impose strict requirements. There are various possible solutions for this serious issue, one of which being implementing retrieval-augmented generation in the models [2, 3]. However, using RAG does not guarantee that there will be no factual incorrectness in the generated text.

Empirical studies have demonstrated that the majority of errors for factual mistakes in text generation using RAG come from the retrieval process [4, 5]. In RAG, the retrieval component assumes that the retrieved documents can be added together into one best answer, however, in practice, the retrieved information conflicts and is inconsistent. This leads to generating factually incorrect text, which is unacceptable in legal corpora, as it leads to serious damage and consequences.

Most research on this topic is about finding ways on improving successful information retrieval using various techniques (re-rankers and chain-of-thought reasoning) [6–8]. However these studies mainly improve how we select context and reason. They do not pay enough attention to identifying and resolving contradictions when different sources have conflicting information, which is often the case in legal context.

Our paper aims to study how RAG systems work in the legal field when there are conflicting statements due to changes over time when regulations or decisions are updated or replaced. We created a set of questions to test the system and see how different ways of finding information affect the answers given and if wrong information shows up due to conflicts. The main contributions of our paper are:

1. An evaluation on how different retrieval strategies affect factual correctness in legal question answering and how they interact with temporal data filtering,
2. An analysis of failures caused by conflicting legal documents,
3. Practical design advice for legal RAG systems based on the performed analysis.

Hypotheses: We test three predictions about legal RAG performance:

1. **H1:** Metadata filtering + hybrid retrieval reduces temporal conflicts (CRR/ORR) >50% vs naive RAG—simple metadata beats complex reasoning.
2. **H2:** Retrieval quality dominates generation errors.
3. **H3:** Hybrid retrieval provides highest ROI: maximum Recall@5 gains at minimum cost increase.

2 Related work

Recent studies on RAG have been centered on discovering ways to improve the precision and reliability of its answers. Most of this research focuses on the individual components of the RAG process and how each of these influences the performance of the system. For example, in [10], the authors compare the performances of a vanilla model, a basic RAG model, and an advanced RAG model with a cross-encoder re-ranker. They find that faithfulness improves from 0.347 with the vanilla model to 0.621 with basic RAG, and up to 0.797 with the advanced setup. However, these studies do not acknowledge legal data or the effects of retrieving documents, which have evolved as time has passed. A similar pattern appears in reinforcement learning–based retrieval approaches. R3, proposed

in the [11] paper, shows a 5% performance improvement, but without testing on temporally dynamic data, it is unclear how effective it would be in legal settings.

Another research direction regarding RAG deals especially with the issue of conflicting documents in RAG systems. The paper [12]presents the RAMDocs dataset, simulating realistic RAG scenarios with ambiguity, misinformation, and noise. The paper also presents the MADAM-RAG model, which is based on the idea of a multi-agent debate on the answers provided by the LLM. On the AmbigDocs dataset, MADAM-RAG achieves up to an 11.4% increase in exact match scores over strong RAG baselines. The model also demonstrates its efficiency in the FaithEval dataset, achieving an improvement up to 15.8%. The Truthful-RAG model [13] is created specifically for dealing with conflicting information. The model is based on the idea of creating knowledge graphs based on the results retrieved by the RAG and then performing query-based graph retrieval with entropy-based filtering to identify conflicts. The results show the efficiency of the model in improving the robustness and trustworthiness of the RAG while ensuring its efficiency. The paper [14] presents the CONFLICTS benchmark, which is annotated with different conflict types in realistic RAG scenarios. The paper demonstrates the inability of LLMs to properly resolve conflicts consistently. Although these approaches and seen trends improve conflict-handling, they do not specifically address outdated or amended legal regulations.

In the legal space, RAG architectures have been tailored to incorporate contradictions and the time component of legal knowledge. The LegalWiz framework [15] utilizes the multi-agent generation framework for detecting and managing contradictions in legal reasoning. It has the capability to automatically generate synthetic legal documents with six different structured forms of contradictions. It is one of the first frameworks for structured contradiction evaluation in the legal RAG space. Another example is Graph RAG for Legal Norms [16], which adapts RAG for legal documents by integrating hierarchical knowledge graphs with richly annotated text segments. This allows the framework to incorporate complex structures and time components of legal knowledge. It has the capability to incorporate internal and external references along with different versions of the regulations. This allows the framework to have richer representations of legal knowledge. The suggested RAG architectures have the potential to improve the accuracy and interpretability of the results in the legal space. However, multi-agent setups and knowledge graphs make systems harder to build, more demanding to run, and more difficult to deploy.

In all, existing works either (1) examine general RAG component ablations without addressing temporal legal conflicts, or (2) design sophisticated architectures for conflict resolution, thereby increasing complexity levels. To the best of our knowledge, there is a lack of research evaluating the influence of standard, practical retrieval components, including dense/hybrid retrieval, metadata filtering by date or legal status, query expansion, and re-ranking, on temporal conflict rates in legal QA systems. In this paper, we address this research gap by focusing on the investigation of temporal conflict resolution, including the ablation of retrieval module components, without the use of any extra layers.

3 RAG System Decomposition

3.1 High-Level RAG Pipeline

The RAG system is composed of three main phases – retrieval, ranking and generation [2, 9]. At a high level, the standard RAG pipeline follows the following stages:

1. Data preparation and indexing: a collection of documents is preprocessed, parted into chunks, embedded and stored in the retrieval component.
2. Retrieval: the RAG system retrieves a set of possible documents from the index based on similarity to the processed question.
3. Context construction: a part of the retrieved content is selected and formatted into a prompt-compatible context.
4. Generation: the large language model generates an output depending on the query and the constructed context.

In some cases there are additional possible phases – query processing and re-ranking or filtering. Query processing happens after data preparation, where the given query can be transformed or expanded, while re-ranking and filtering happens after retrieval, when the received information is re-ranked or filtered using additional models, methods. Both of these stages are introduced to try and improve the overall retrieval-generation process.

3.2 Responsibilities of Individual Components

The data preparation and indexing phase decides what information is available for the whole RAG system. The decisions made in this phase (the chunk size, overlap, indexing approach) can seriously affect how the system will find relevant context. If mistakes are made at this point, we might not be able to get the significant documents needed to answer a query accurately.

Secondly, the retrieval phase of the RAG system is responsible for efficiently identifying relevant and significant documents. For this part there are various different methods that can be chosen – dense retrieval, sparse retrieval or a hybrid approach utilizing both retrievers. At the end of retrieval the system produces a candidate dataset of relevant documents, which is then later used in all following stages.

Next, the context construction stage decides what information to show the language model and how to format it. This includes deciding how many pieces of information to include, the order of the documents and if any of them need to be shortened.

The final stage is generation, which uses the information it had been given to produce the final response. It is responsible for ensuring that the generated answer is fluent and linguistically correct, however factual accuracy is not guaranteed at this stage.

3.3 Where Errors Originate

Errors in the RAG system occur at different phases of the RAG pipeline [4]. The most common stages where mistakes happen are the retrieval, context construction and generation phases. Retrieval failures happen when the correct documents are not found or they are ranked too low. Context construction errors arise when the RAG system is provided with incomplete or poorly formatted information. Lastly, generation errors occur because the language model does not interpret context correctly or fails to reason properly when generating the final text.

4 Retrieval strategies

Retrieval strategies determine how relevant documents are selected from the knowledge base before being provided to the language model. In this section, we present several retrieval mechanisms commonly used in RAG systems, including sparse, dense, and hybrid retrieval approaches. Additionally, we examine complementary techniques such as query rewriting, metadata filtering, and reranking that can improve retrieval quality and reduce temporal conflicts in legal documents.

4.1 Sparse retrieval

Sparse retrieval algorithms are based on keyword matching between queries and documents. Traditional sparse retrieval algorithms like BM25 estimate document relevance based on the number of keyword matches. The top documents are then sent to the LLM.

4.2 Dense retrieval

Dense retrieval uses neural embedding models to convert user queries and available documents into vectors. After such transformation, texts with similar semantic meaning are positioned closely together. We use FAISS for efficient similarity search. In our implementation, document chunks are embedded and stored in the FAISS index during the indexing phase. At retrieval time, user query is embedded using the same method and matched against the vectors to return the top-k most relevant documents.

4.3 Hybrid retrieval

Hybrid retrieval combines the two previous approaches and executes both of them in parallel. It then merges the results using a fusion algorithm. This method combines keyword matching with semantic similarity search.

We perform hybrid retrieval by concurrently performing dense vector search with FAISS and sparse keyword search with the BM25 algorithm. To combine the two, we use a weighted sum fusion technique that normalizes results based on defined weights. In this case, we assign equal weights of 50% to dense and sparse search algorithms.

4.4 Query rewriting

Query rewriting is a method of transforming the original query issued by the user into new expressions that better match the system's requirements.

HyDe (Hypothetical Document Embedding) is one of the query rewriting methods. It produces a hallucinated response to the given query and then relies on this response to retrieve relevant documents before displaying the final result to the user.

Manual or templated rewriting is another query rewriting approach, considered to be one of the simplest query modification techniques. In essence, a predetermined template is used which is populated with a given user query. In our system, we use the following templates:

- *Decision appointing {X}*,
- *Decision amending or repealing previous act concerning {X}*,
- *Decision establishing requirements, conditions or framework for {X}*,

where X represents the original user query.

4.5 Metadata filtering

Metadata filtering can improve RAG systems by refining search queries. Leveraging this technique, RAG system adds selected metadata to a search query (e.g., date, status, etc.) filtering out irrelevant documents early in the search process. Thus, it is an irreplaceable tool in large and complicated datasets.

In our experiments, two metadata filters are evaluated. The first is a temporal filter that restricts retrieval to documents published after a specified year (e.g., date $>$ 2010). The second is a legal status filter that limits results to documents marked as *In Force*, ensuring that only currently valid legal acts are considered during retrieval.

4.6 Re-ranking

Re-ranking in RAG system means reordering a retrieved set of documents according to their relevance to the original query. This step helps to improve the quality of information that the LLM receives to generate a response.

We use the Cross-Encoder model from the Sentence Transformers library. Unlike other retrieval models, which compute embeddings for queries and documents separately, the Cross-Encoder computes the embedding for the query-document pair in a single forward pass. This allows the model to capture more complex semantic relationships that are not captured by embedding-based similarity metrics.

The re-ranking procedure has three steps. Firstly, each retrieved document pairs with the original query and passes through the Cross-Encoder, which produces a refined relevance score. Secondly, candidate documents sort in descending order by these scores. Lastly, the reordered list and updated relevance values replace the initial retrieval ranking.

5 Experimental Scope & Data Preparation

This section describes the experimental scope of the study and the preparation of the data used in our experiments. First, we explain how the dataset was constructed and how the final subset of legal acts, queries, and ground-truth annotations were created. Next, we present the chunking strategy and the embedding and indexing procedures used to build the retrieval infrastructure. Finally, we describe the experimental setup by distinguishing between fixed and variable components of the RAG pipeline.

5.1 Dataset Construction

For this paper, the EurLex dataset was used [17]. The dataset contains 142036 legal European Union acts passed between 1952–2019. Each law has a CELEX, various legal metadata, such as act type, date, status, attributes for amendment relationships and the full law's text.

The first stages of the dataset construction involved downloading the full dataset, preprocessing it and performing an initial inspection. The preprocessing process included normalizing CELEX identifiers for each law, converting the date format and parsing amendment chains. As for the initial inspection, its purpose was to investigate five chosen acts and see where conflicts between them arise. The analysis demonstrated that CELEX identifiers with numeric suffixes are not versions or amendments of the same law – thus to find amendments or conflicts between legal decisions, we had to focus on explicit amendment relationships and other temporal legal metadata.

After completing working on the full EurLex dataset, we moved on to forming the final data subset. The subset was created using metadata filters and a selection process involving different steps. In the end, the final dataset was constructed, which contains 555 legal acts, where 231 acts are acts with amendments, 229 acts that are expired and 326 active acts.

The last step of the dataset construction process was creating the query and ground-truth sets. For the query set, 100 legal acts were randomly selected from the previously created subset of 555 documents while making sure that each chosen legal decision comes only from one of the four buckets: acts with amendments, expired acts, active acts and legal acts in general. Then for each law a factual and temporal question was written - this formed the basis of the query set. As for the ground-truth dataset, we took the created questions from the query set and assigned to each the corresponding law's CELEX, the expected date range, and the difficulty level (easy, medium or hard).

5.2 Chunking Strategy

The chunking strategy for this paper involved chunking only the created EurLex subset of 555 legal documents. The text was split by sentences into chunks of 512 tokens, with a 32-token overlap between chunks. Each chunk captures all relevant information associated with the legal acts – CELEX, the act's status, publication

date, etc. Chunks are stored in dense FAISS embeddings and additionally in a sparse BM25 index, where hybrid retrieval is chosen (see section 3.4).

5.3 Embedding and Index Construction

For this paper we have two choices for retrieval type - dense or hybrid retrieval. For dense retrieval, each chunk is embedded using the OpenAI's text-embedding-3-small model, which produces 1536-dimensional vectors. These vectors are then stored in a FAISS IndexFlatL2 index, which supports efficient nearest-neighbor search over high-dimensional embeddings [18]. We save the constructed index to disk and we have the choice to rebuilt it from the same dataset if needed. In the cases where hybrid retrieval is used, we combine the FAISS index with a sparse BM25 index built over the same chunks. BM25 uses token-level term matching and supports optional top-k filtering, while FAISS provides semantic similarity [19].

5.4 Fixed vs. Variable Experimental Components

Since our paper performs different experiments in order to investigate how the RAG system works when conflicts in documents are present, this setup leads to some components being fixed and some – variable. Fixed elements remain constant across all performed experiments: the dataset, overall RAG pipeline architecture created using LlamaIndex, chunking parameters, index structures, the embedding model and the language model. While the variable components are those which differ depending on the different defined experiment variants. Variable components include retrieval type (dense or hybrid), top-k retrieval, metadata filtering (constraints on document date), query expansion, and re-ranking using a cross-encoder model. Query expansion has two possible choices in our setup – manual rewriting or HyDe-generated reformulations. The detailed setup for all experiment variants can be found in Table 1.

Table 1. Experiment matrix

ID	Retrieval	Top-k	Query Expansion	Filters	Re-ranking
V0	Dense	5	None	None	No
V1	Dense	10	None	None	No
V2	Hybrid	5	None	None	No
V3	Dense	5	None	Date (>2010)	No
V4	Hybrid	5	None	Status (In Force)	No
V5	Dense	5	Manual rewrite	None	No
V6	Dense	5	HyDE	None	No
V7	Dense	5	None	None	Yes
V8	Hybrid	5	None	None	Yes
V9	Hybrid	5	HyDE	Date (>2010)	Yes

6 Evaluation Methodology

In this paper, for effective evaluation of how RAG deals with legal data, where conflicts or law amendments are present, we focus on both the retrieval and generation phases of the pipeline. We aim to identify retrieval strategies that reduce outdated answers, ensure that answers are complete and avoid introducing new reasoning failures. In order to achieve this goal, both retrieval and generation-level metrics are tracked and later analyzed. In addition, we discuss cost and latency considerations in order to understand the computational efficiency of the evaluated pipelines. Finally, we outline the reproducibility guarantees applied in our experiments to ensure that results can be consistently reproduced.

6.1 Retrieval-level metrics

The retrieval-level metrics focus on the documents returned by the RAG system before the answer generation process. Since our work focuses on version conflicts and temporal misalignments, the chosen metrics indicate which regulatory versions are retrieved and what temporal information the model can use when generating the outcome. The metrics we decided to use in our paper are conflict retrieval rate (CRR), outdated retrieval rate (ORR), conflict density (CD) and latest version recall (LVR).

CRR measures the proportion of retrieved CELEXs that participate in at least one amendment relationship within the retrieved set. CRR is calculated by the following formula:

$$CRR = \frac{C}{R},$$

where C is the subset of retrieved regulations that participate in at least one amendment relationship with another regulation in the retrieved set R. Higher CRR means that more of the retrieved documents are linked through amendments, so the model is more likely to encounter different versions of related regulations. On the other hand, lower CRR means the retrieved set focuses on fewer related versions, making the context easier to interpret.

CD shows how many of the retrieved regulations are connected to each other through amendments, compared to the overall size of the retrieved set. To measure the CD metric we use the formula:

$$CD = \frac{P}{R},$$

where P is the set of retrieved regulations pairs that are connected by amendments. A high CD value means that many of the retrieved regulations are linked to each other by amendments compared to the overall number of retrieved documents. Meanwhile, a low CD value shows that few amendment relationships exist among retrieved documents.

ORR measures the proportion of retrieved CELEX identifiers that are considered outdated. A CELEX is considered outdated if its status metadata marks

it as such or if its valid time period ended before the reference date. The ORR is computed with the formula below:

$$ORR = \frac{O}{R},$$

where O is the set of retrieved outdated CELEX identifiers. A high ORR value leads to the fact that a large share of retrieved documents are no longer valid at the reference date. In contrast, a low ORR means that most of the retrieved documents are still valid.

LVR measures whether the latest version of a regulation is present in the retrieved set. We calculate this metric in two ways: the first situation assumes that the relevant CELEX is known. Given this case, we calculate LVR by the following formula:

$$LVR = \begin{cases} 1, & \text{if latest}(c^*) \in R \\ 0, & \text{otherwise} \end{cases}$$

where c^* is the expected CELEX, latest(c^*) is the latest version in the knowledge base and R is the set of retrieved CELEX identifiers. If LVR equals to 1 that means that the system retrieved the most recent version, while 0 indicates that the latest regulation was not retrieved and generating a factually accurate answer is unlikely.

The second situation assumes that the relevant CELEX is not given, then to measure LVR we use the formula:

$$LVR = \frac{|\{\, c \in R \mid \text{latest}(c) \in R \,\}|}{|R|}$$

If this formula is used then a high LVR value means that most retrieved CELEX's include their latest versions and a low LVR indicates that the retrieved documents often lack their most recent version.

6.2 Generation-level metrics

Generation-level metrics measure how the model behaves when given the retrieved context. They are analyzed to understand how the model reasons and generates the final output when conflict is present in the given information. The generation-level metrics we chose to track in our paper are answer temporal alignment (ATA), citation consistency (CC), hedging behavior (HB) and answer completeness (AC).

ATA classifies which regulation version the answer reflects. We calculate this metric by first identifying all the regulations mentioned in the generated answer and then comparing each one to the most recent version in our knowledge base. Based on the results, we classify the generated output as latest, outdated, mixed or unclear. A latest output demonstrates that the model used up-to-date

information to generate its response. On the other hand, answers which are labeled outdated or mixed indicate that the model relied on old or conflicting information.

CC measures whether the citations in the final answer are up-to-date and correctly match the documents that were retrieved. We calculate CC by first listing all regulations cited in the answer, then comparing these citations to the set of retrieved documents to see if they were actually provided and checking if multiple versions of the same regulation are cited. Lastly, the answer is classified into four of the available classes: clean, conflicting (multiple versions of the same regulation are cited), outdated-only and no-citation. An output labeled as clean means all citations are current and appear in the retrieved documents. Conflicting, outdated and ungrounded outputs suggest the answer could be misleading.

HB measures the certainty of the model's produced answer. To calculate this metric we search for words or phrases that indicate uncertainty, such as, "might," "could," "typically", or conditional statements ("if", "depends on"). Then using the extracted information, the outputs are classified as confident, hedged and conditional (valid only under certain circumstances). AC measures how fully the answer addresses the question, regardless of whether it is correct. We measure this metric by counting the words in the answer and compare to predefined minimum thresholds. Next, we look at the important terms in the question and see how many of them are mentioned in the answer. Lastly, based on the previous steps, the produced output is classified as non-answer, partial and complete. Complete outputs fully answer the query, partial outputs – address only some parts and non-answers are insufficient.

6.3 Cost and latency tracking

In order to evaluate a RAG system effectively it is insufficient to only track metrics related to accuracy, mentioned in sections 6.1 and 6.2. It is also very important to measure cost and latency, since LLM queries very quickly can become expensive and slow generation can be impractical, especially in time-sensitive workflows.

In our paper, we measure computational cost in two ways. The first counts tokens in the input and output using a standard tokenizer, then calculates the LLM cost by multiplying these token counts by their per-thousand-token rates. The second looks at embedding tokens, accounting for the cost of creating embeddings for the retrieved documents.

As for latency, in our work we do not automatically measure latency. Nonetheless, tracking response times is still important for practical deployment. Latency could be measured by recording timestamps at each stage of the pipeline, including retrieval, embedding, and generation.

As for latency tracking – we do not automatically measure latency, as we do not plan to deploy our created RAG system. Nonetheless, latency tracking is still an important measure that could identify where the system is too slow or gets stuck. Latency could be measured by recording timestamps at each stage of the pipeline, including retrieval, embedding, and generation.

6.4 Reproducibility guarantees

In this work, reproducibility was guaranteed wherever it was possible. For instance, we used a fixed seed for the random selection of documents that were used to create the final query and evolution sets (see Section 5.1). Additionally, some components used in our experiments remained fixed, so that differences seen in our results could be better explained by the variable components (see Section 5.4). Although the model's answers can still vary because of its natural randomness, keeping the environment consistent helps ensure that the retrieved documents, token counts, and key generation metrics stay consistent across experiments.

7 Results

This section presents the experimental evaluation of the proposed legal RAG system variants. First, we analyze the impact of individual system components through a series of ablation experiments that isolate retrieval strategies, filtering mechanisms, query expansion, and re-ranking techniques. We then perform a failure analysis to better understand the sources of incorrect outputs and distinguish between retrieval-bound and generation-bound errors. Together, these analyses provide insight into how different design choices influence the factual correctness and reliability of legal RAG systems.

7.1 Component Ablation Experiments

Ten different variants of the study, from V0 to V9, were used, each modifying a single component to isolate its effect. Each variant was evaluated on a similar corpus of 200 EurLex legal queries (except V5: 198 queries), and the metrics measured were Recall@5, non-answer rate, average coverage, citation consistency, ungrounded citation rate, outdated retrieval rate, and cost per query.

Our performed experiments revealed that increasing the top-k retrieval window from 5 (V0) to 10 (V1) had no impact on Recall@5, which remained at 1% because the metric considers only the first five retrieved documents. The non-answer rate decreased from 6.5% to 2.5%, but the overall correctness did not improve. In addition, token usage and cost nearly doubled, making this adjustment an inefficient trade-off (Table 2, Figure 1).

Additionally it was seen that switching from dense-only retrieval (V0) to hybrid retrieval (V2), which combines dense embeddings with BM25, resulted in a substantial performance increase. Recall@5 improved from 1.0% to 82.5%, the non-answer rate dropped to 0.5%, and coverage increased to 90.8%, while ungrounded citation rates remained similar (see Table 3, Figure 2).

Applying metadata filters demonstrates significant trade-offs between recall, outdated retrievals, and clean citations. Dense retrieval with a date filter (V3) reduced outdated retrievals but nearly eliminated recall (0.5%) and doubled the ungrounded citation rate to 39.5%, with no clean citations. Hybrid retrieval with

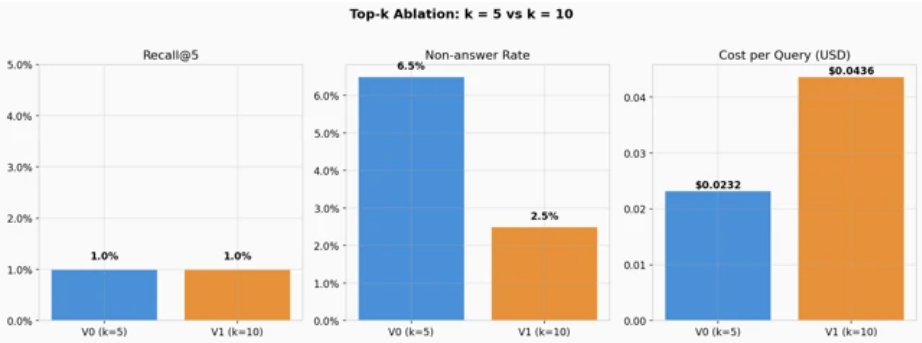

Fig. 1. Top-k ablation: k = 5 vs. k = 10

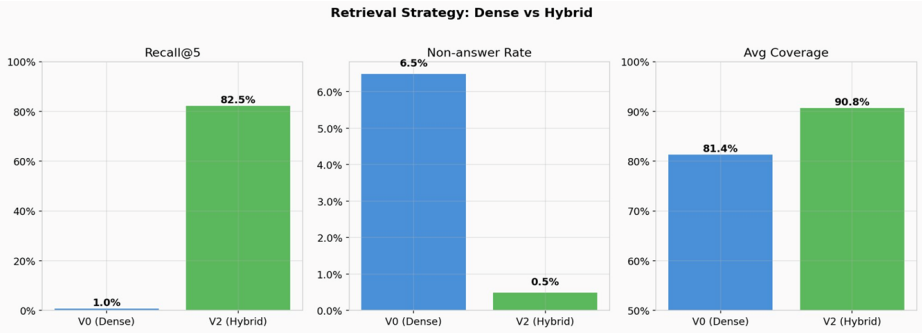

Fig. 2. Retrieval strategy comparison

Table 2. Top-k selection ablation (V0 vs V1)

Metric	V0 ($k = 5$)	V1 ($k = 10$)	Delta
Recall@5	1.0%	1.0%	0 pp
Non-answer rate	6.5%	2.5%	-4.0 pp
Avg. coverage	81.4%	87.1%	+5.7 pp
Ungrounded citation rate	13.5%	13.5%	0 pp
Cost per query (USD)	0.0232	0.0436	+88%
Total prompt tokens	405,267	800,387	+97%

Table 3. Retrieval strategy ablation (dense vs hybrid)

Metric	V0	V2	Delta
Recall@5	1.0%	82.5%	+81.5 pp
Non-answer rate	6.5%	0.5%	-6.0 pp
Avg. coverage	81.4%	90.8%	+9.4 pp
Ungrounded citation rate	13.5%	14.0%	+0.5 pp
Cost per query (USD)	0.0232	0.0255	+10%

a status filter (V4) eliminated outdated retrievals entirely and resulted in the highest number of clean citations (107 out of 200), but reduced Recall@5 to 43.5% due to the exclusion of ground-truth documents outside the filter window (see Table 4, Figure 3).

Table 4. Metadata filtering ablation

Metric	V0	V3	V2	V4
Recall@5	1.0%	0.5%	82.5%	43.5%
Non-answer rate	6.5%	3.5%	0.5%	4.5%
Outdated retrieval rate	60.4%	32.0%	56.1%	0.0%
Ungrounded citation rate	13.5%	39.5%	14.0%	18.5%
Clean citation count	16/200	0/200	58/200	107/200
Cost per query (USD)	0.0232	0.0075	0.0255	0.0133

The experimentation also showed that query expansion, using either manual rewriting (V5) or HyDE generation (V6), did not improve Recall@5, which remained at 1.0%. Manual rewriting reduced the non-answer rate from 6.5% to 2.0% and increased coverage, but also increased cost. HyDE slightly decreased the ungrounded citation rate to 10.0% and incurred a fixed generation cost, but without hybrid retrieval, these advantages were limited (see Table 5, Figure 4).

The introduction of a cross-encoder re-ranker (V7 and V8) did not affect Recall@5 but improved secondary metrics, including non-answer rate and ungrounded citations. The cost difference between re-ranked and non-re-ranked

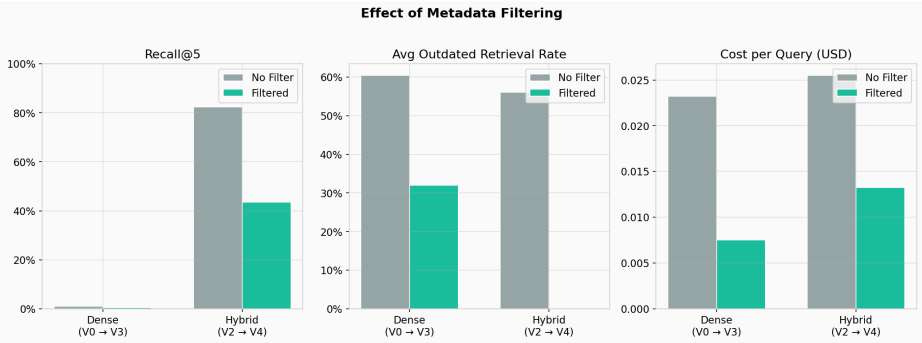

Fig. 3. Impacts of metadata filtering

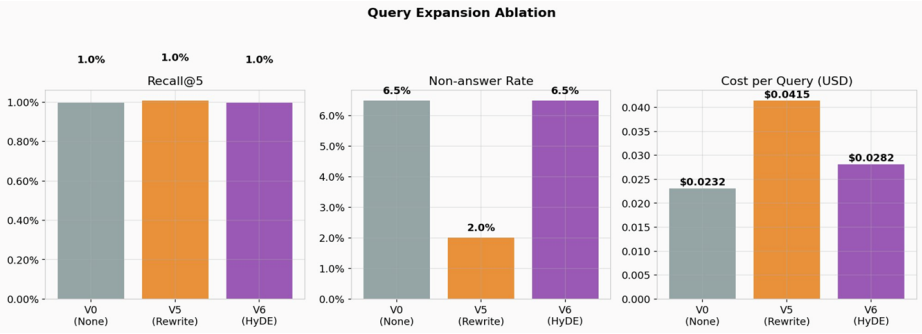

Fig. 4. Query expansion ablation

Table 5. Query Expansion Ablation

Metric	V0	V5	V6
Recall@5	1.0%	1.0%	1.0%
Non-answer rate	6.5%	2.0%	6.5%
Avg. coverage	81.4%	87.5%	79.9%
Ungrounded citation rate	13.5%	14.1%	10.0%
Cost per query (USD)	0.0232	0.0415	0.0282
HyDE generation cost	None	None	0.9862

variants was negligible. This demonstrates that re-ranking is most useful when a sufficient pool of candidate documents is available (see Table 6, Figure 5).

Table 6. Re-ranking Ablation Results

Metric	V0	V7	V2	V8
Recall@5	1.0%	1.0%	82.5%	82.5%
Non-answer rate	6.5%	5.0%	0.5%	1.0%
Ungrounded citation rate	13.5%	11.0%	14.0%	12.5%
Avg. coverage	81.4%	81.4%	90.8%	89.4%
Cost per query (USD)	0.0232	0.0230	0.0255	0.0254

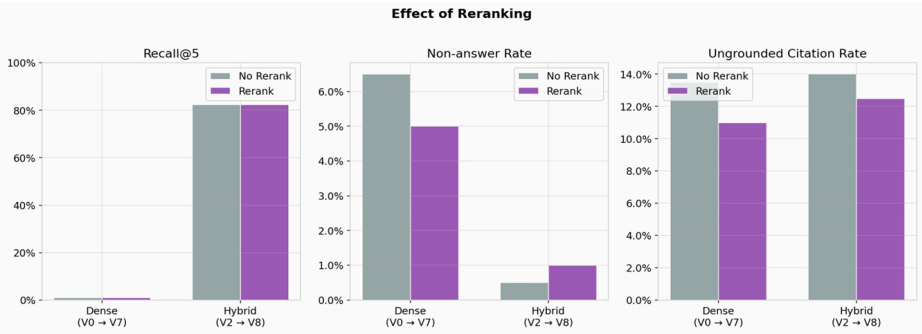

Fig. 5. Impact of re-ranking

The complete pipeline (V9) achieved a Recall@5 of 34.0%, reduced cost per query by 28%, but increased the ungrounded citation rate to 25.0% (see Table 7, Figure 6). Overall, the experimentation proved that hybrid retrieval combined with re-ranking provides the most effective balance between recall, cost, and citation quality.

Table 7. Combined pipeline vs naive baseline (V9 vs V0)

Metric	V0	V9	Delta
Recall@5	1.0%	34.0%	+33.0 pp
Non-answer rate	6.5%	2.5%	-4.0 pp
Avg. coverage	81.4%	87.8%	+6.4 pp
Ungrounded citation rate	13.5%	25.0%	+11.5 pp
Outdated retrieval rate	60.4%	49.2%	-11.2 pp
Cost per query (USD)	$0.0232	$0.0168	-28%
HyDE generation cost (total)	None	$0.9952	–

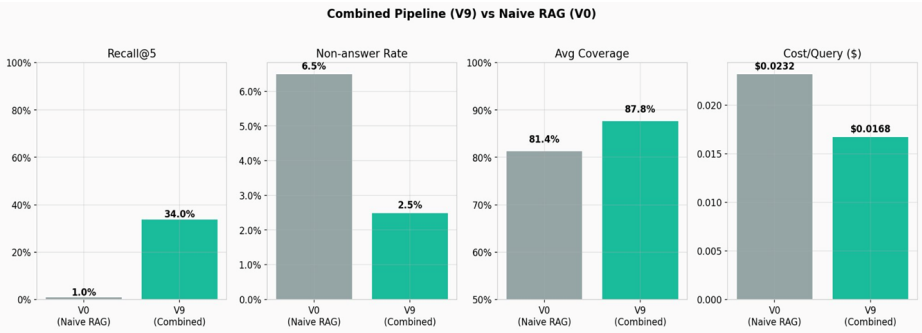

Fig. 6. Combined pipeline (V9) vs. naive RAG (V0)

7.2 Failure analysis

This section is dedicated for investigating the seen failures in our performed experiments across different RAG variants. We classify the observed failures as retrieval-bound and generation-bound. Retrieval-bound errors occur when the system fails to find the correct or latest law, while generation-bound failures happen when the correct law is retrieved, but the answer is incomplete, temporally misaligned, or contains citation mistakes. If neither of these errors are detected for an output, the answer can be classified as "success" or "partial success". Successful answers require retrieving the latest law with fully correct, complete, and properly cited laws, while outputs that are mostly correct but not perfect are labeled as partial success. The detailed requirements for these four answer classes can be found in Table 8.

After classifying all the answers we were able to see the distributions of each category in different contexts (see Figure 7). From the figure's first graph we observe that the majority of outputs were classified as retrieval-bound failures – they contain around 78% of all answers. This result proves the seen trend in previous papers, that most RAG errors happen at the retrieval phase. As for generation-bound failures, they account for around 10.96% of all generated answers. Successful answers have a very similar count as generation-bound failures and the minority class is the partially successful answers, which have a total of 3

Table 8. Requirements for failure type classification

Answer Type	Requirements
Retrieval-bound failure	$LVR = 0$ or ($ORR = 1$ and $ATA \in$ {Ambiguous/Unclear, Outdated})
Generation-bound failure	$LVR = 1$ and ($AC =$ Non-answer or $ATA \neq$ Latest or $CC \neq$ Clean)
Success	$LVR = 1$ and $ATA =$ Latest and $CC =$ Clean and $AC =$ Complete
Partial success	Any answer not meeting the strict success criteria but not classified as retrieval- or generation-bound failure.

answers. If we were to look at the second graph, which presents the answer type counts based on question types, we would see that the overall distributions are similar, although temporal questions have led to more failures compared to the factual queries. Lastly, looking at the third graph we can analyse seen trends based on the question's difficulty level. For different difficulty levels we observe that retrieval-bound errors largely exceed all other answer types. Additionally, it can be seen that for medium and hard questions generation failures are the second most popular category, while for easy questions – successful answers, which could have been anticipated, as they should be more easily answered.

Moreover, after successful answer type classification, we analysed retrieval-bound and generation-bound failure causes (see Figure 2). We found that, for retrieval errors, most mistakes (82.9% of all retrieval failures) come from LVR = 0, while for generation errors – the leading causes are citation issues (66% of all generation failures) and ATA being ambiguous or outdated (33.4% of all generation failures).

Looking at the answer type distributions per variants (see Figure 9 and Table 9), we can see that the worst performing variants are V0, V1, V3, V5, V6 and V7, as they do not produce any successful or partially successful answers and all their outputs are classified as retrieval errors. The rest variants show more promising results, with V4 demonstrating the best result (successful answers outnumber the rest of the categories). Additionally, it can be seen that there are no cases where generation-bound failures exceed retrieval-bound errors. The mean CRR and ORR values seen in the table reinforce these findings. Variants dominated by retrieval failures tend to show relatively high ORR values, which indicate frequent reliance on outdated information. In contrast, V4, the strongest variant, has both mean CRR and mean ORR equal to zero, suggesting that it effectively avoids conflicting and outdated retrieval.

Finally, Tables 10, 11, 12, and 13 provide concrete examples for each answer category. We find that for the retrieval-bound failures, the latest law wasn't retrieved (LVR = 0), which by itself causes misclassification, even if the answer is otherwise complete (see Table 10). In contrast, the generation errors retrieve the correct law (LVR = 1), but still produce answers that are outdated or not properly supported (see Table 11). Table 12 shows that in partial success cases,

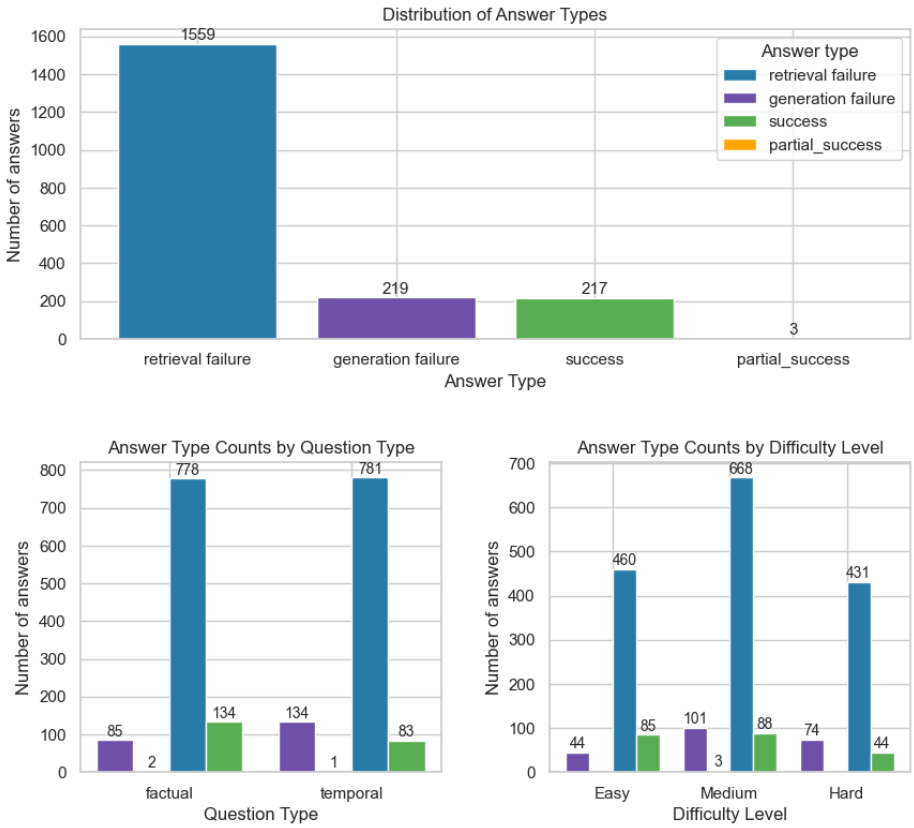

Fig. 7. Answer type distribution results

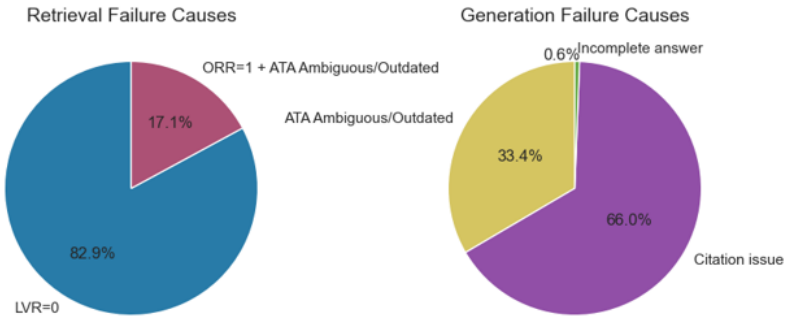

Fig. 8. Retrieval-bound and generation-bound failure causes

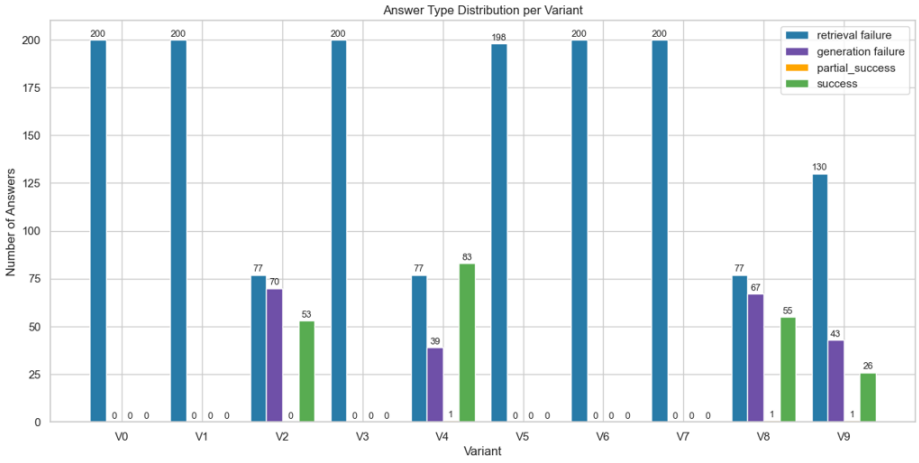

Fig. 9. Answer type distribution per variant

Table 9. Failure results summary per variant

Variant	Failure %	Retrieval-bound %	Generation-bound %	Mean CRR	Mean ORR
V0	100.0	100.0	0.0	0.00	0.60
V1	100.0	100.0	0.0	0.02	0.56
V2	73.5	38.5	35.0	0.18	0.56
V3	100.0	100.0	0.0	0.00	0.32
V4	58.0	38.5	19.5	0.00	0.00
V5	100.0	100.0	0.0	0.01	0.55
V6	100.0	100.0	0.0	0.02	0.66
V7	100.0	100.0	0.0	0.00	0.60
V8	72.0	38.5	33.5	0.18	0.56
V9	86.5	65.0	21.5	0.18	0.49

the correct law is retrieved and citations are accurate, but minor timing or completeness issues remain. As for the successful examples, seen in Table 13, we find that they satisfy all strict criteria, demonstrating the system behaving as expected.

Table 10. Retrieval-bound failure examples

Variant	Question	LVR	ATA	ORR
V6	Before the 1996 amendment, how was "capon" defined for the purpose of EU marketing standards?	0.0	Ambiguous/Unclear	0.50
V6	What conditions must operators meet to receive agricultural payments in euros instead of their national currency?	0.0	Ambiguous/Unclear	1.00

Table 11. Generation-bound failure examples

Variant	Question	LVR	ATA	CC	AC
V2	What were the models used by Member States for statements of accounts for entitlements to own resources before this 2016 Decision?	1.0	Outdated	Outdated-only	Complete
V8	What was the main purpose of negotiating amendments and additions to the Annexes and Protocol of the Agreement on fishing off the coast of Senegal?	1.0	Outdated	Outdated-only	Complete

Table 12. Query examples which produced partially successful answers

Variant	Question	LVR	ATA	CC	AC
V9	Which EU procedures for VAT administrative cooperation were applied around October 2003?	1.0	Mixed	Clean	Complete
V8	Which EU office provides a permanent secretariat to support the Panel in its work?	1.0	Latest	Clean	Partial

Table 13. Query examples which produced successful answers

Variant	Question	LVR	ATA	CC	AC
V2	How were categories of beef for intervention determined prior to February 23, 2001?	1.0	Latest	Clean	Complete
V2	Before 2018, how were critical functions, core business lines, and financial interconnections reported and mapped?	1.0	Latest	Clean	Complete

8 Practical implications

Our results indicate that the quality of the information retrieved is the main factor that affects the reliability of legal RAG systems. As shown in the failure analysis, the majority of errors (78%) correspond to retrieval-bound failures, which occur when the system fails to retrieve the appropriate or latest version of the legal act. Therefore, the main focus in the development of legal RAG systems should be the improvement of the system's retrieval capacity.

In our experiments, we see that the variants that combined dense and sparse retrieval, as well as filtering legal metadata, resulted in the best performance regarding system reliability. Specifically, we see that the hybrid approach resulted in the largest gain in recall with minimal increase in cost, thus indicating that this method has the highest "return on investment" regarding implementation effort. Conversely, we observed that variants that relied solely on dense retrieval were unable to retrieve the correct document. There are some interesting trade-offs that we see from our ablation study. Increasing the window size, or top-k, had a large impact on cost but resulted in minimal improvements to retrieval accuracy. Metadata filtering was seen to be important for preventing outdated results, but overly stringent metadata filters resulted in reduced recall by preventing relevant historical results from being included. Similarly, query expansion techniques, such as manual query rewriting and HyDE, were seen to be ineffective if the underlying retriever was already ineffective. Re-ranking resulted in small improvements to citation grounding, but only when relevant documents were already included in the candidate set.

Overall, we believe that, prior to the implementation of more sophisticated approaches, the focus should be on the improvement of the system's capacity for information retrieval. Improvements such as hybrid retrieval and carefully tuned metadata filtering can already provide substantial gains in reliability with relatively low implementation complexity.

9 Limitations & future work

While this research offers important insights into the function of different components in the retrieval process that influence the management of temporal conflicts

in legal RAG systems, there are several limitations that provide a direction for future research. First and foremost, the experiments conducted in the present research are based only on a subset of the EurLex dataset, which contains legal acts from the European Union. However, the effectiveness of the various retrieval strategies proposed in the present research may not be effective for different legal databases.

Second, the filtering techniques investigated in this study have been based on static metadata constraints. Although these techniques proved to be useful for eliminating stale documents, they may not be appropriate for dealing with the time complexity issue caused by legal documents, where the relevance of a legal regulation may depend on the time context of the query. Future research could investigate dynamic temporal filtering strategies that adjust retrieval constraints during the search process. These methods would allow metadata filters to adapt to the query's intent and any temporal cues present in the question.

Another limitation of our study is the simplicity of the architecture used in the evaluated system. While our research focused on assessing the efficacy of RAG systems with more practical retrieval configurations and without additional layers of reasoning, recent studies have shown that more advanced architectures, such as multi-agent debate or knowledge graphs, could potentially help in resolving conflicts and improving the reliability of RAG systems in retrieving information.

Finally, although the evaluation of the RAG system was conducted in terms of retrieval quality, time correctness, and reliability of citations, other aspects of legal reasoning, like interpretability of legal arguments or even consistency in relevant regulations, were not considered in this evaluation. Future work could focus on the extension of the evaluation metrics in order to better understand the effect of the retrieval strategies on the accuracy and the usability of legal AI systems.

10 Conclusion

10.1 Evaluation Setup

In this paper, an exhaustive evaluation of ten different variants of the legal RAG system was successfully performed to analyze the efficiency of different retrieval strategies and the accuracy of their generated answers. The effectiveness is assessed using latest version recall, conflict retrieval rate, outdated retrieval rate, answer temporal alignment, citation consistency, answer completeness, and the cost of the operation, among other factors. The RAG system's variants are evaluated using 200 created queries from EurLex subset.

10.2 Key Findings

These ablation experiments offered useful insights into the performance of the legal RAG system:

- Increasing the query window size by doubling the value of k, changing it from 5 to 10, slightly increased non-answer rates, but the recall rates remained the same, and the costs almost doubled.
- Hybrid retrieval had the best overall results, with the Recall@5 value increasing from 1.0% to 82.5%, while also reducing non-answer rates and improving answer coverage at a relatively low cost.
- Metadata filters were effective in reducing the number of outdated results, although overly selective filters reduced the overall recall and non-answer rates, particularly with historically framed queries. Notably, some configurations (e.g., V4) demonstrated that lower recall can still lead to better temporal correctness and citation quality.
- Query expansion, both manual and using HyDE, had little effect on overall recall, although the HyDE results improved citation rates with dense-only approaches.
- Re-ranking with cross-encoders showed some positive results, improving non-answer and citation rates, but these results are not sufficient to mask the failure of the first and most important stage of the overall system.
- Using all of the components (in this study, version 9), the Recall@5 value was increased from 1.0% to 34.0%, with reduced costs per query, although recall rates were reduced due to strict application of temporal filters compared to the application of hybrid retrieval without aggressive filters.

Furthermore, the failure analysis shows that most errors (78%) are retrieval-bound, mainly due to not retrieving the latest version of a regulation (LVR = 0). This suggests that retrieving correct and up-to-date documents is more important than recall alone. In addition, higher recall can still include outdated results (high ORR), so it should be considered together with temporal and citation-based metrics.

10.3 Recommendations

In summary, the results show that having correct metadata brings the most to the table in terms of mitigating retrieval failures and temporal conflicts. However, we understand that this is not always possible given large unstructured datasets, which are commonly used in knowledge bases for RAG systems.

- For these cases, we recommend sticking to hybrid as the default approach, as it provides the best trade-off between recall, temporal correctness, and overall answer reliability.
- We also advise against using re-ranking and HyDe techniques until the retrieval-level issues are addressed, as they improve citation and non-answer metrics but do not improve retrieval correctness.

References

1. Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

2. Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in neural information processing systems 33 (2020): 9459-9474.
3. Gao, Yunfan, et al. "Retrieval-augmented generation for large language models: A survey." arXiv preprint arXiv:2312.10997 2.1 (2023).
4. Zhang, Wan, and Jing Zhang. "Hallucination mitigation for retrieval-augmented large language models: a review." Mathematics 13.5 (2025): 856.
5. Vach, Marius, et al. "Evaluating Retrieval Augmented Generation-enhanced Large Language Models for Question Answering On German Neurovascular Guidelines." Clinical Neuroradiology (2025): 1-9.
6. Karpukhin, Vladimir, et al. "Dense passage retrieval for open-domain question answering." Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). 2020.
7. Xu, Jian, et al. "RALLRec: Improving retrieval augmented large language model recommendation with representation learning." Companion Proceedings of the ACM on Web Conference 2025. 2025.
8. Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837.
9. Cheng, Mingyue, et al. "A survey on knowledge-oriented retrieval-augmented generation." arXiv preprint arXiv:2503.10677 (2025).
10. Maharjan, Anuj, and Umesh Yadav. "Chunking, Retrieval, and Re-ranking: An Empirical Evaluation of RAG Architectures for Policy Document Question Answering." arXiv preprint arXiv:2601.15457 (2026).
11. Zhou, Jiawei, and Lei Chen. "Optimizing Retrieval for RAG via Reinforcement Learning." arXiv preprint arXiv:2510.24652(2025).
12. Wang, Han, et al. "Retrieval-augmented generation with conflicting evidence." arXiv preprint arXiv:2504.13079 (2025).
13. Liu, Shuyi, Yuming Shang, and Xi Zhang. "TruthfulRAG: Resolving Factual-level Conflicts in Retrieval-Augmented Generation with Knowledge Graphs." arXiv preprint arXiv:2511.10375 (2025).
14. Cattan, Arie, et al. "Dragged into conflicts: Detecting and addressing conflicting sources in search-augmented llms." arXiv preprint arXiv:2506.08500 (2025).
15. Mantravadi, Ananya, et al. "LegalWiz: A Multi-Agent Generation Framework for Contradiction Detection in Legal Documents." arXiv preprint arXiv:2510.03418 (2025).
16. de Martim, Hudson. "Graph rag for legal norms: A hierarchical and temporal approach." arXiv e-prints (2025): arXiv-2505.
17. EurLex Dataset. "EUR-Lex Dataset." Kaggle. Available at: https://www.kaggle.com/datasets/puskas78/eurlex-dataset
18. "Faiss Documentation." Available at: https://faiss.ai/index.html
19. Robertson, Stephen, and Hugo Zaragoza. "The probabilistic relevance framework: BM25 and beyond." Foundations and Trends in Information Retrieval 3.4 (2009): 333–389.