Linking Rationale to Decision on Internet Standards: A Retrieval-Based Approach Using Synthetic Data

Jie Bian, Michael Welzl
Department of Informatics, University of Oslo
{jiebi, michawe}@ifi.uio.no

Abstract

The Internet Engineering Task Force (IETF) develops Internet-Drafts (I-Ds) and Requests for Comments (RFCs) as formal specifications for Internet Protocols. While these documents capture finalized technical standards, the rich design rationales and deliberations that shape them are often buried in informal discussions across mailing lists. These discussions are rarely linked explicitly to the specifications they inform, making it difficult to trace the origins of specific design decisions. We address this gap by generating synthetic data that explicitly links discussion threads to their corresponding RFC/I-D sections, producing roughly 350 000 such aligned instances. This data enables training a semantic embedding-based information retrieval (IR) system that, given an email discussion, retrieves the most relevant specification content. Our experiments show that this synthetic supervision helps models learn associations between informal discourse and formal documentation, though the task remains challenging due to the implicit and context-dependent nature of the links.

Keywords: RFC, Email, Synthetic Data, Information Retrieval

1. Introduction

The Internet Engineering Task Force (IETF) defines the protocols and standards that power the modern Internet (McQuistin et al., 2021; Morabito and Jiménez, 2020; Lin et al., 2023). A key part of this process is its extensive use of mailing lists (Welzl et al., 2021; Khare et al., 2022) for collaboration and decision-making. These mailing lists capture technical discussions, working group (WG) activities (Bradner, 1998), and meeting outcomes, forming a rich historical record of Internet development. Today, the IETF maintains nearly 40 years of archived mailing list history, dating back to its early days in the mid-1980s.

The IETF begins by developing Internet-Drafts (I-Ds) (Farrel, 2014), which serve as preliminary versions of technical specifications. Through iterative revisions and community review, these drafts may eventually be published as Requests for Comments (RFCs). RFCs, the formal documents that define these standards, are highly technical documents intended for specialists. They often rely on dense terminology and assume significant background knowledge, making them difficult for outsiders to interpret. This complexity creates barriers on multiple fronts, e.g., educators and students often struggle to understand the concepts for teaching and learning purposes, or practitioners and developers must invest considerable time and effort to interpret RFCs when implementing protocols. In this context, the IETF mailing list archive[1] is an invaluable resource. Primarily conducted in English, it preserves the discussions, debates, and design rationales behind those standards defined in RFCs, offering critical context often absent from the formal specifications. Access to this historical record enables the stakeholders to trace the evolution of internet technologies and better understand the decisions that shaped them.

However, navigating the IETF mailing list archive is far from easy. First, the archive is massive and continuously growing, making it time-consuming to locate a specific email or thread related to a particular technical detail. Second, the content of these emails is often highly context-dependent. They are written for participants actively involved in drafting Internet standards, which makes them difficult for outsiders to follow. Even for individuals who are not complete outsiders, such as members from other WGs, the content of these discussions can appear opaque and difficult to interpret. Without the shared context of prior exchanges, it is often unclear which specific technical details are under consideration. The authors sometimes include excerpts from the relevant RFC or I-D to provide context. More often, however, they reference only the location within a document, e.g., “Section 7.1, second paragraph, last bullet point,” or no reference is provided at all, as the participants in the exchange are presumed to share the necessary background knowledge. Therefore, despite the wealth of information contained within the IETF mailing list archive, it remains significantly underutilized.

The recent shift toward open, version-controlled workflows has also influenced the IETF, with an increasing number of WGs adopting GitHub as a collaborative platform for drafting I-Ds and managing contributions (Cooper and Hoffman, 2020).

[1] https://mailarchive.ietf.org

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 7149–7162
11–16 May 2026.

Building on this trend, Bian et al. (2025) constructed a dataset capturing the relationship between design rationales (discussions) and design decisions (RFC edits) from these repositories. This was accomplished by leveraging GitHub artifacts like issues, which document rationale through discussion, and pull requests (PRs), which implement decisions through proposed edits. These artifacts are often explicitly linked, as PRs typically reference the issues they resolve, creating a structured connection between the rationale expressed in discussions and the decisions reflected in edits.

Building on the compiled dataset, Bian et al. (2025) introduced retrieval tasks (Karpukhin et al., 2020) designed to leverage the linkage between rationale and decision. It defined two directions: rationale-to-decision, which retrieves relevant edits based on discussion content, and decision-to-rationale, which identifies associated discussions for a given edit. These tasks aim to enhance traceability and support design understanding. However, the dataset was limited in scale, which constrains the robustness of such retrieval tasks. Moreover, applying similar methods to email-based workflows remains challenging, as email lacks the explicit artifact linkage present in GitHub, leaving email discussions largely underutilized for rationale–decision analysis.

To address these challenges, we propose a two-step framework that combines synthetic decision reconstruction and context-aware IR. First, we leverage LLMs to reconstruct the decision implied in an email discussion, generating a text snippet that corresponds to the relevant section of the associated RFC or I-D. This is accomplished through prompt-based generation, where the input rationale guides the model to produce a decision-like output. This step compensates for missing or incomplete design decisions in email-based workflows. Second, we use the reconstructed decision together with the original email discussion (rationale) to create training pairs for an IR model.

Our IR task is defined as *rationale-to-decision* retrieval, where the input is email discussions and the target is the corresponding decision. As a pseudo example, the rationale "to simplify implementation across platforms, we avoided using platform-specific cryptographic primitives" would lead to the decision "It SHOULD use standardized AES encryption with a fixed key length". While our IR system is symmetric, using the same language model (LM) to encode queries and documents, and thus capable of supporting either retrieval direction, we focus on *rationale-to-decision*. This choice reflects a practical consideration: many decisions in RFC development lack explicit discussion records and are often derived from authors' prior knowledge or established practices. Evaluating *decision-to-rationale* would penalize the system for missing evidence rather than retrieval quality. Using *rationale-to-decision* maximizes observable ground truth and yields more reliable metrics.

We position this as a *first-of-its-kind* effort to leverage email discussions as a bridge to formal specifications, transforming a previously underutilized resource into a valuable tool for internet standards navigation. This approach serves as an intermediate step toward formalizing the rationales behind Internet Protocols. Email discussions are typically expressed in free-form, often comprising fragmented, informal, and context-dependent exchanges. By retrieving and aligning email (threads) with corresponding decisions, we enable the construction of more structured and formal explanations (Lewis et al., 2020), aligned with the rationale outlined in RFC 8374 (Morrow et al., 2018). See our public codebase for implementation details on synthetic data generation, model fine-tuning, and evaluation.[2] In summary, our contributions are as follows:

- Design Decision Recovery: We introduce a specialized prompting strategy that enables LLMs to infer and reconstruct missing design decision from technical email discussions.
- **RFCAlign**: A Semi-Synthetic Rationale-to-Decision Dataset: We present RFCAlign, the first dataset designed for rationale-to-decision mapping in the domain of networking protocol design. RFCAlign comprises around 350 000 samples spanning 63 IETF WGs. The dataset is semi-synthetic: rationales are sourced from the existing IETF mailing-list archive, while the associated decisions are synthesized. The curated dataset is publicly available on HuggingFace.[3]
- **RFC-DRAlign models**: Using RFCAlign, we fine-tune Mistral-7B (Jiang et al., 2023) for rationale–decision (R–D) retrieval, establishing a new benchmark for this task. The resulting models are released on HuggingFace.[4] Additional variants are available under the same account.

2. Related Work

Synthetic Training Data Generation is widely applied in both natural language and domains where labeled data is scarce, costly, or sensitive, such

[2] https://github.com/cheop-byeon/mteb-R2Gen

[3] https://huggingface.co/datasets/jiebi/RFCAlign

[4] https://huggingface.co/jiebi/RFC-DRAlign-QN

as Code (Xu et al., 2025; Nadăș et al., 2025), Mathematics (Yu et al., 2024; Lu et al., 2024), Science (Taylor et al., 2022; Li et al.), Law (Huang et al., 2023; Yue et al., 2023) and Medical (Bao et al., 2023; Xu et al., 2024). Prompting techniques are central to LLM-driven synthetic data generation, from zero-shot, one-shot, and few-shot to controlled methods using task-specific constraints, templates, or iterative refinement (Chai et al., 2025; Lee et al., 2024).

Information Retrieval Retrieval models within the Beir (Thakur et al., 2021) MTEB (Muennighoff et al., 2022; Enevoldsen et al., 2025) benchmarks demonstrate robust capabilities. Their success stems from their ability to encode queries and documents into aligned vector spaces, facilitating accurate similarity matching. Beyond text embedding–focused approaches, recent research has shifted toward retrieval models with enhanced reasoning capabilities; Notable work include Bright (Hongjin et al.), the first text retrieval benchmark that requires intensive reasoning to retrieve relevant documents, and ReasonIR (Shao et al., 2025), the first retriever purpose-built for broad reasoning tasks.

3. Design Decision and Rationale

For better illustration, we showcase examples of design decisions and rationales drawn from real-world GitHub data, formal RFC documentation, and our synthetically generated instance in Figure 1.

3.1. Design Decision

Formally, we define *design decisions* in I-Ds or RFCs as the normative and descriptive statements within an Internet standard that specify the chosen mechanisms, structures, and operational behaviors of a protocol. These decisions represent the outcome of a selection process among possible alternatives, constrained by functional requirements, interoperability goals, and architectural principles.

3.2. Design Rationale

We define *design rationale* as a set of explanatory statements that account for why specific textual elements in an I-D or RFC appear in their current form. These statements may reflect design trade-offs, rejected alternatives, or historical and contextual motivations. RFCs rarely include such rationales by design, as their primary purpose is to serve as concise, implementable specifications rather than detailed records of the decision-making process. As a result, much of the reasoning behind protocol structures exists outside the RFC itself.

Rationale (represented as discussion):
In our interop testing, all of the implementations of portals require that the DHCP option is in the request list in order for it to be sent. We should be explicit that this is what SHOULD be done in the document (clients SHOULD request).

Decision: Clients that support the captive portal DHCP option should include the option in the parameter request list in DHCPREQUEST messages. DHCP servers may send the captive portal option without any explicit request.

Rationale (derived from email thread):
In the current BGP protocol, any AS can withdraw, at any time, any prefix it previously announced. The rationale for not signing withdrawals is that BGPsec assumes the use of transport security between neighboring BGPsec routers. Thus, no external entity can inject an update that withdraws a route or replay a previously transmitted update containing a withdrawal...

Decision: Withdrawals are not signed.

Rationale (summarized by LLM from an email):
The document reports a technical errata in RFC9000, the specification for QUIC (Quick UDP Internet Connections), a secure transport protocol. The errata concerns Section 8.1.2, which describes how a server should handle a client Initial packet with an invalid Retry token. The reported issue is that the current text does not clearly specify how to handle tokens that are unreadable or generated by another server, leading to potential connection closure in multi-CDN contexts. The proposed correction scopes the section to only apply to tokens that are correctly formatted and readable by the server, but whose contents are insufficient to prove the client's transport address is valid.

Decision (synthesized by LLM):
If a server receives a client Initial containing a valid Retry token that does not authenticate the peer address, the token SHOULD be ignored and the packet handled as if no token were present, as specified in Section 8.1.3, to ensure compatibility with multi-CDN contexts where tokens generated by one CDN may be received by a different CDN.

Figure 1: The top example represents a real-world data sourced from GitHub related to I-D 7710bis, later published as RFC 8910 (Kumari and Kline, 2020). The middle example, drawn from RFC 8374 (Morrow et al., 2018), illustrates an ideal case in which the design rationale for RFC 8205 (Lepinski and Sriram, 2017) is explicitly articulated by the author through concluding from email discussion threads. The bottom example showcases synthetic data generated using Llama-70B from email discussions within the QUIC WG

Rationale and Decision Dynamics An initial draft is normally first presented to a WG at a meeting, and new drafts usually prompt further discus-

sion, e.g., on whether a new draft version resolves an issue raised earlier. Especially controversial decisions spark intensive discussions and exchanges of ideas among the WG members. Then these proposals, critiques, and consensus outcomes directly shape the I-D updates. In such adaptive design processes, where decisions are continuously revised, design rationales and decisions form a co-evolutionary cycle: rationales capture and justify current choices while shaping future revisions, and decisions in turn update the rationale space. This mutual evolution helps maintain traceability and coherence as both elements iteratively inform one another.

Genres of Design Rationales We analyzed RFC 8374 (Morrow et al., 2018), which, to our knowledge, is the only document summarizing key design decisions and discussions that informed the development of RFC 8205 (Lepinski and Sriram, 2017). Based on this analysis, we gained a clearer understanding of the specific rationales involved in the RFC decision-making process and subsequently identified and categorized the following genres of rationale:

System Design: These rationales reflect considerations related to technical constraints and operational feasibility in the overall system architecture.

Performance Optimization: This category includes decisions aimed at minimizing overhead, such as reducing the size of data to be hashed in order to avoid performance penalties.

Risk Mitigation and Security Validation: These rationales focus on ensuring the absence of security vulnerabilities in the proposed design. They emphasize the importance of addressing potential failure modes and validating the robustness of the security model.

Pragmatic Trade-offs: These decisions involve balancing competing priorities, such as maintaining efficiency without compromising core security guarantees, or navigating trade-offs between privacy and security, and between performance and compatibility.

4. Data Source

4.1. Synthetic Data Generation Source: WGs and Mailing Lists

The GitHub repositories examined in Bian et al. (2025) are associated with 63 WGs. We retrieved the corresponding email communications for these WGs from the IETF mail archive to analyze their contextual relevance. For each WG, we obtained all available emails from the group's inception up to the date of collection, using the ietfdata toolkit.[5]

We extracted both the subject line and message body from each email and concatenated them to construct the full email text. To support analyses at varying levels of verbosity, we applied two preprocessing strategies. In the **verbose** version, we preserved the entire email content, including quoted text from previous messages. While the **non-verbose** version excluded quoted segments, typically identified by lines beginning with the character >, to isolate the sender's original contribution. Given that verbose emails often contain richer contextual information, we utilized them for generating the corresponding design decisions. Both versions were utilized as training data for IR.

4.2. Platform Differences

While many IETF WGs have traditionally relied on email for communication, some have increasingly adopted GitHub in recent years for drafting and collaboration (Cooper and Hoffman, 2020). In practice, WGs now often use **both** mailing-list and GitHub. *Some WGs, especially new emerging ones, have shifted to using GitHub as their primary collaboration platform*, where most technical discussions and decision-making occur within issues and pull requests. Consequently, these WGs tend to have significantly lower volumes of email communication compared to those that rely more heavily on mailing lists.

For other WGs, mailing-list remains the primary communication channel. A key distinction between email and GitHub lies in the structure and depth of the discussions. Email threads are typically longer and more detailed, often comprising multi-paragraph messages that delve into complex technical reasoning and broader contextual analysis. This format supports extended deliberation, particularly on intricate or contentious topics. In contrast, GitHub issue comments are generally shorter and more focused, addressing specific implementation tasks or incremental changes.

We observed instances of cross-referencing between email and GitHub discussions. For example, GitHub issue comments sometimes include links to relevant mailing list threads, often with remarks like "this was discussed, see the email at [URL]." Similarly, emails reference specific GitHub issues by including direct links. These references are typically concise and serve to bridge related conversations across platforms without duplicating their content. Our analysis shows that the substantive discussions on each platform are largely distinct.

[5] https://github.com/glasgow-ipl/ietfdata/

4.3. Annotated Email Data

We build on two previously annotated RFCs, RFC 7657 (July 2014 to November 2015) and RFC 8335 (June 2016 to February 2018), in which email messages were manually aligned with corresponding RFC text segments, as introduced in Bian et al. (2024). In this study, we extend the dataset by incorporating a new dataset RFC 8205 (Lepinski and Sriram, 2017), by extracting rationale–decision pairs formally documented in RFC 8374 (Morrow et al., 2018), an Informational companion to RFC 8205 that documents its (8205) key design rationales and decisions without introducing new mechanisms. Collectively, these three annotated RFCs (7657, 8335, and 8205) serve as a real-world test set for evaluating methods that retrieve rationale-to-decision derived from email discussions.

5. Synthetic RFCAlign Generation

Since email discussions often contain the rationale but lack explicit statements of the final design decisions, we use synthetic data to fill in the missing decision content for training. Specifically, we prompt an LLM with an email message and instruct it to infer the design decisions discussed within that message. This task formulation aligns with prompt-based data augmentation techniques, where carefully constructed prompts, potentially including task-specific instructions or illustrative examples, are used to guide LLMs in generating new data instances (Nadăș et al., 2025). Unlike traditional QA generation (Shakeri et al., 2020; Puri et al., 2020), which creates questions from a given context, our approach reverses the direction: the email serves as an "question", and the model generates the corresponding "context" (i.e., the design decision), resembling a reverse QA generation paradigm.

5.1. Direction: Synthetic Design Decision and/or Synthetic Rationale

Our study aims to retrieve rationale (email discussions) and decision (I-Ds/RFCs) pairs, and then generate formal rationales (Lewis et al., 2020) which are structured and high-level summaries that justify the decisions. Thus, it is essential to construct high-quality training pairs, the decision, and its corresponding ground truth explanation. While synthetic data generation offers scalability, fully artificial pairs often lack the realism needed for effective learning. A more viable approach is to use real data for one component (either the rationale or the decision) and generate the other, preserving authenticity while enabling model training.

This framing highlights a key challenge: generating explanations for real decisions is inherently as difficult as the original task itself, making it impractical for data augmentation. To validate this, we conducted preliminary experiments in which we provided an LLM with pairs of modified RFC text segments (before and after) and asked it to explain the changes. The model predominantly produced generic editorial justifications (e.g., "improved clarity" or "fixed formatting") rather than capturing the underlying design rationale. It suggests that deriving rationales from decisions is particularly challenging, as it often requires broader context, such as prior discussions, stakeholder concerns, or historical design trade-offs, that are not present in the RFC text alone.

Given the difficulty of generating rationales from decisions, we adopt the inverse strategy: generating plausible decisions from real rationales. This approach leverages the relative simplicity of the inverse task and aligns with established practices in synthetic data generation (Shakeri et al., 2020; Puri et al., 2020), where constructing inputs for known outputs often yields more realistic and informative training data. Email discussions frequently contain explicit proposals, technical reasoning, and references to intended changes, offering strong cues for reconstructing the corresponding decisions. Moreover, this direction supports practical traceability needs, as stakeholders often begin with informal discussions and seek to understand the resulting formal decisions. For these reasons, we focus on generating design decisions from email discussions to produce **RFCAlign**, rationale–decision(synthetic) pairs.

5.2. LLM for RFCAlign Generation

We leverage LLMs for RFC design decisions generation, as their pre-training exposes them to a wide range of internet protocols and RFCs. We select two open-source LLMs, namely Qwen-32B[6] and Llama-70B,[7] as generators. These models demonstrate strong instruction-following capabilities, which are critical for adhering to structured prompts in our pipeline, as well as their proven performance on text generation tasks (Jindal et al., 2024). Qwen-32B offers competitive accuracy with moderate resource requirements, while Llama-70B provides a larger parameter space for capturing nuanced technical language. Specifically, we generate our data using Meta's tool Synthetic Data Kit[8] and vLLM (Kwon et al., 2023),[9] which together

[6] https://huggingface.co/Qwen/Qwen2.5-32B-Instruct

[7] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

[8] https://github.com/meta-llama/synthetic-data-kit

[9] https://github.com/vllm-project/vllm

enable scalable and efficient prompt-based generation. Table 5.2 provides an overview of the datasets analyzed in the study, including both previously published datasets from Bian et al. (2025) and our newly created dataset, each broken down into relevant subsets. Note that we do not count unique queries or documents in the training set.

Source	WG	I-Ds	Split	Pairs	U.R.	U.D.
GitHub	63	160	train	4 196	-	-
			dev	1 221	539	1 221
			test	1 299	521	1 297
		41	ood	4 307	1 534	4 063
Email	63	-	train (QS)	342 765	-	-
		-	train (LS)	379 859	-	-
	1	1	test (RFC 7657)	163	105	39
	1	1	test (RFC 8205)	46	46	46
	1	1	test (RFC 8335)	72	45	30
	3	3	test (RFCs)	281	196	115

Table 1: Dataset overview. Abbreviations: U = unique, R = rationale, D = decision, S = synthetic, Q = Qwen, L = Llama, I-Ds = Internet-Drafts. OOD (out-of-domain) refers to a subset drawn from WG repositories different from those used for the train, development, and test splits. "–" indicates ignored values. RFCs comprise three WGs: dart (7657), sidr (8205), and intarea (8335). The full list of WGs used for the synthetic data can be found in Table 6 in the appendix.

5.3. Prompts

In our preliminary experiments, we designed three prompting strategies to explore different levels of guidance for generating RFC design decisions from email discussions. These prompt levels vary in the degree of constraint and structure they impose on the LLM. We adopt a zero-shot generation, where the model receives no example outputs, only the input rationale.

Try-Then-Infer Strategy The model is instructed to produce an RFC-style design decision directly from the email content. This setting provides the model with considerable interpretive flexibility, enabling it to infer and articulate a plausible decision based on the discussion. However, it also introduces variability, leading to outputs that may lack precision or exhibit overly generalized phrasing.

Judge-Then-Infer Strategy We introduced a more structured way. The model is first asked to determine whether an email contains substantive technical discussion or normative content, as opposed to non-technical messages such as social interactions, administrative notices, or automated system messages. Then, for emails deemed relevant, the model is tasked with generating a corresponding text snippet that mirrors the formal tone and terminology typical of RFCs or I-Ds. If the email includes a direct quotation from a specification, the model is instructed to preserve it; otherwise, it synthesizes a plausible segment that aligns with the conventions of RFC standards documents.

Anchor-Then-Infer Strategy We explicitly incorporate the email's subject line into the prompt as supplementary context. Subject lines frequently reference specific Internet-Drafts or RFC identifiers (e.g., RFC 1234), which serve as strong indicators of the relevant specification. This additional signal was expected to help the model constrain its generation to the correct document, thereby improving its ability to produce RFC-style text snippets that accurately reflect the technical content discussed in the email. However, the model often failed to produce satisfactory outputs, contrary to expectations.

We attribute this to several factors: First, the subject line provides minimal semantic information beyond the identifier, which leads to an implicit assumption that the model can reconstruct the associated technical details from its pre-trained knowledge. Second, the model may overfit to the identifier without leveraging the email body effectively, and the inherent ambiguity of mapping informal discussion to formal specification text remains unresolved. Last but not least, we underestimate the complex and often overlapping relationships among RFCs, where a single document may reference multiple prior specifications. This high degree of interconnectedness introduces ambiguity, making it difficult for the model to determine the exact context or target RFC constraint based on the draft name shown in the subject line.

After evaluating the outcomes of preliminary experiments with different prompt designs, we determined that **Judge-Then-Infer Strategy** yielded more consistent and accurate results. We therefore adopted this strategy; see Figure 2 for details. While post-processing (filtering out low-quality generated data after generation) is common (Chai et al., 2025), our second strategy involves pre-filtering, prompting the model to decide whether the email contains substantial technical or regulatory content. If the email is deemed irrelevant, such as an administrative message or social exchange, the model simply outputs an empty string. Defining a reliable filtering or scaling mechanism for post-processing is inherently difficult because our task requires reasoning-aware alignment rather than mere surface similarity. RFC decisions and email rationales differ in structure, tone, and intent, and their relationships are often many-to-many. This complexity makes scoring schemes inadequate.

6. Rationale to Decision Retrieval

6.1. Task Formulation

Design decisions in Internet standards are not made arbitrarily, and they are typically the result of underlying *design rationales*, statements that justify, explain, or motivate the choices embedded in the specification. Specifically let:

- $\mathcal{D} = \{\delta_1, \delta_2, \ldots, \delta_n\}$ be the set of design decisions. (as defined in Section 3.1),
- $\mathcal{R} = \{r_1, r_2, \ldots, r_m\}$ be the set of design rationales, (as defined in Section 3.2),

To capture the relationship between rationales and decisions, we define a mapping:

$$f : \mathcal{R} \rightarrow 2^{\mathcal{D}} \tag{1}$$

where $f(r_j) \subseteq \mathcal{D}$ denotes the subset of design decisions influenced by rationale r_j. We develop a retrieval systems that, given a rationale r_j, aim to identify the corresponding set of decisions $f(r_j)$.

6.2. Baseline Approach

This task demands both deep semantic understanding and inferential reasoning, particularly when queries and relevant documents exhibit minimal lexical similarity or only shallow semantic alignment. Following conventions, we explored traditional retrieval methods BM25 (Robertson et al., 2009), and modern LM-based approaches (Karpukhin et al., 2020; Muennighoff et al., 2022). For dense retrieval (Karpukhin et al., 2020), we selected a diverse set of models varying in scale, capabilities, and architectural focus, ranging from those optimized for semantic similarity (Weller et al., 2024) to those designed for reasoning-intensive retrieval (Shao et al., 2025).

Beyond on the models examined in Bian et al. (2025), such as GritLMs (Muennighoff et al., 2024) and Promptrievers (Weller et al., 2024), contributing to the MTEB benchmark (Muennighoff et al., 2022), we extend our evaluation with models tailored to reasoning-intensive tasks. This is motivated by our observation discussed in Section 3.2 that rationales often follow identifiable logic patterns. Specifically, we incorporate specialized architectures including ReasonIR-8B (Shao et al., 2025), optimized for inferential retrieval; Nemotron-8B (Chen et al., 2025), designed for multi-step reasoning; and Qwen-7B (Team, 2024), known for its strong contextual understanding.

6.3. Baseline Models and Backbone Model for Our Task

We adopt BM25 (Robertson et al., 2009) as the baseline for sparse retrieval. For dense retrieval, we evaluate the above models, both in their original configurations and after fine-tuning on the I-Ds GitHub train split (Bian et al., 2025). Evaluations are conducted on the GitHub development and test sets. Our results indicate that reasoning-oriented models underperform compared to those optimized for semantic similarity. This may be due to a mismatch between the type of reasoning required in our task, as analyzed in Section 3.2, and the reasoning capabilities these models were trained on (Math, Code, etc). Alternatively, it indicates that deep reasoning is not essential during the first-stage retrieval for the data, but may play a more beneficial role in following tasks such as reranking (Karpukhin et al., 2020) or explanation generation (Lewis et al., 2020).

Among the semantic similarity-focused models, Promptriever, built on Mistral-7B, achieves the best performance. Consequently, we select it as our dense retrieval baseline and adopt Mistral-7B (Jiang et al., 2023) as the backbone for further fine-tuning on our curated dataset.

We also include the SIGIR model (Bian et al., 2025), trained on the I-Ds GitHub training split, as a baseline representative of models developed from real-world data. This enables direct comparison with our model trained exclusively on synthetic data.

6.4. Synthetic Training Data

We train retrieval models using the synthetic data and assess its quality through downstream retrieval performance. If a model trained solely on synthetic examples generalizes well to real evaluation data, this indicates that the synthetic pairs capture the relevant signals needed for the task. In our study, we assess whether synthetic email–design-decision pairs can fulfill the role of real annotated pairs for training retrieval models, particularly when authentic supervision is limited or unavailable.

6.5. Test Data

We use three manually annotated email datasets, RFC 7657, RFC 8205, and RFC 8335, and their combined version (RFCs) to represent email data, while also including the GitHub test and OOD sets to comprehensively evaluate both our baselines and aligned models. For GitHub data and the combined RFCs set, the IR system searches for relevant decisions across all; for individual RFC datasets, the search is restricted to the corresponding WG.

6.6. Training Details

SFT (supervised fine-tuning) We train our model using a contrastive loss with in-batch negatives (Karpukhin et al., 2020), with batch size 24

and 2 GPUs. Each instance consists of one positive pair and 47 negative pairs, 23 sampled from the same device and 24 from a different device. The input query (email) has a maximum length of 4096 tokens (verbose), and 3584 (non-verbose), while the document (design decision) is up to 512 tokens. We use a learning rate of 1e-4 and apply parameter-efficient fine-tuning (PEFT) (Mangrulkar et al., 2022), updating only approximately 1% of the model's full parameters.

Model Selection Due to the large volume of synthetic training data, we trained the model for only one epoch and saved checkpoints every 100 steps, resulting in a total of eight checkpoints. To ensure a fair comparison, we used this GitHub dev split to evaluate the checkpoints. Since this dev set is derived from another distribution (GitHub), the selected checkpoint may not necessarily be optimal for the email-based data. We employed two LLMs in conjunction with two distinct email processing strategies, mentioned in Section 4.1, resulting in four fine-tuned model variants: (1) QN: qwen-non-verbose, (2) QV: qwen-verbose, (3) LN: llama-non-verbose, and (4) LV: llama-verbose.

7. Result and Analysis

7.1. Metrics

We use MRR@k as our primary evaluation metric because the task focuses on how highly the model ranks the first relevant decision. Although some rationales may correspond to multiple valid decisions, our current objective is to retrieve at least one correct item, for which MRR is sufficient; more fine-grained metrics such as NDCG will be considered in future work. We report MRR@10 for GitHub and MRR@5 for Email to match their candidate-set sizes. All MRR values are scaled by 100.

7.2. Baseline Models

BM25 is robust in our task, while Promptriever underperforms on our Email dataset, likely because it struggle with highly structured, domain-specific content like RFCs, where rigid terminology and minimal semantic variation limit the benefits of its pre-training from extensive IR task datasets.

7.3. Performance on GitHub Data

GitHub Results (MRR@10) Table 2 reports MRR@10 on the GitHub dataset. The retriever trained on real-world GitHub data, **Mistral**^{R} (we use this notation for better comparison; R represents the real world), achieves the best performance on both the in-domain test and the OOD split, as expected given the shared distribution. On both splits, **Mistral**^{R} achieves higher performance than **Mistral**^{S}. Both learned retrievers outperform BM25 and Promptriever, with **Mistral**^{R} offering the most significant improvements. This makes it the preferred model for GitHub data.

GitHub	BM25	P	M^R	M^S (mean$\pm$sd)
Test	63.23	67.76	**79.27**	71.93 $\pm$ 1.51
OOD	41.49	46.71	**57.16**	48.24 $\pm$ 1.47
Macro-avg	52.36	57.24	**68.22**	60.09

Table 2: MRR@10 on GitHub data. M^R: Mistral retriever trained on *real* GitHub data; M^S: same architecture trained on *synthetic* data; we report mean$\pm$sd across four S variants (QN, QV, LN, LV); Full per-variant results are in Table 4 in the Appendix.

7.4. Performances on Email Data

Table 3 compares model performance across different RFC test sets. Among these, RFC 8205 is the highest-quality subset, manually analyzed to yield concise rationales and a clear one-to-one mapping to design decisions; RFC 7657 and RFC 8335, by contrast, contain decisions distributed across multiple emails and emails that reference several decisions. Here, the email-based rationale is typically diffuse and embedded in extended discussion, complicating the mapping.

Email	BM25	P	M^R	M^S (mean$\pm$sd)
RFC 7657	54.33	60.60	60.30	**61.25** $\pm$ 2.26
RFC 8205	73.91	70.83	**81.96**	80.73 $\pm$ 1.84
RFC 8335	56.74	46.41	46.52	**61.13** $\pm$ 6.48
RFCs	60.35	59.23	62.22	**65.60** $\pm$ 1.75
Macro-avg	61.33	59.27	62.75	**67.18**

Table 3: MRR@10 on email data. M^R: Mistral retriever trained on *real* GitHub data; M^S: same architecture trained on synthetic data; we report mean$\pm$sd across four S variants (QN, QV, LN, LV). Full per-variant results are in Table 5 in the Appendix.

Email results (MRR@5) Relative to GitHub, the email benchmarks are more heterogeneous. **Mistral**^{S} leads on nearly all email subsets except RFC 8205, and consistently outperforms BM25 and Promptriever. Overall, **Mistral**^{R} ranks second. The performance gap between **Mistral**^{S} (first) and **Mistral**^{R} (second) is small on RFC 8205 and RFC 7657, but **Mistral**^{S} shows a substantial lead on RFC 8335. Because the distinction between the two models is not always clear-cut, we conduct a qualitative case analysis in Section 7.5 to examine these effects in detail.

Sparse BM25 vs. Dense Promptriever Emails combine conversational and technical material and

are typically much longer than GitHub comments, providing BM25 with a richer set of lexical cues to match against. Meanwhile, important information in emails is often dispersed across replies, headers, and quoted fragments from earlier messages, which can dilute the signal within a single embedding and make dense retrievers less decisive when ranking closely related candidates.

Qwen vs. Llama Empirically, training on Qwen-generated data, particularly the non-verbose (QN) variant, **Mistral**^{S} (QN), yields the best retrieval scores overall, whereas manual inspection suggests that Llama's outputs often read closer to RFC style. This contrast indicates that *stylistic fidelity to RFC prose is not, by itself, the key driver of retrieval effectiveness*. Instead, properties such as lexical clarity, concise phrasing, and well-structured presentation of salient anchors (identifiers, error terms, actionable statements) appear to align better with our retriever. In other words, synthetic data that foregrounds the relevant cues the model is trained to exploit can outperform more stylistically faithful text.

7.5. Case Study

We define a top$k$ failure, or miss, as the model failing to place the relevant decision within the top$k$ results (i.e., hit@k equals 0). We conduct case studies on **Mistral**^{R} and **Mistral**^{S} (QN) using RFC 8205 and RFC 8335, examining top 10 failures.

- RFC 8205: **Mistral**^{S} (QN) has 3 failures; **Mistral**^{R} has 4, with 3 shared; **Mistral**^{R} has 1 extra unique miss. We present the additional 1 miss in Figure 3.
- RFC 8335: **Mistral**^{S} (QN) has 8 failures; **Mistral**^{R} has 9, with 8 shared; **Mistral**^{R} has 1 extra unique miss. We present the additional 1 miss in Figure 4.

RFC 8205 The rationale is strongly tied to Decision 1 (gold) because its core proposal is to outsource BGPsec path validation (and possibly signing) to an off-router box that returns a verdict before routes are placed in the Adj-RIB-In. Most of the text is a risk/benefit analysis of that outsourcing model, e.g., how much of the control plane is offloaded, the reliability of the external validator and its communications, and centralized vs. distributed deployment, i.e., the exact considerations that justify delegating cryptographic validation/signing off the router.

Mistral^R conflates *where* validation occurs (placement/policy in Decision 2) with *how* validation is executed (Decision 1's off - board/outsourced validator–signer). This collapses the rationale into a gist like "validate at the ingress edge and forward," while overlooking the decisive architectural cues anchoring Decision 1: delegation to an off-router box, obtaining a verdict before admitting routes to the Adj-RIB-In, and the outsourcing-specific failure modes.

RFC 8335 The rationale focuses on ICMP Code registries, registration policy, and explicitly references First Come First Serve (FCFS), along with questions about per-registry code ranges (e.g., "maximum values for the Code registries ... 15?"). This content clearly aligns with Decision 1. The model's selection of Decision 2 as the top result can be interpreted as Decision 2 essentially repeating the background description found at the end of the email.

Based on the observed failure patterns, we infer that **Mistral**^{R} prioritizes broad semantic similarity over explanatory alignment, which leads to over-reliance on surface phrasing and missed rationale signals. In contrast, **Mistral**^{S} (QN) appears to better adapt to the variety of rationale types present in emails, likely due to the greater complexity and larger size of the email data.

8. Limitation

We evaluate the model in an idealized setting where each rationale comes from a curated set that is known to align with one or more decisions. This enables controlled testing of the model's ability to retrieve associated decisions. Yet this evaluation does not reflect real-world retrieval: operationally, the entire email archive serves as the rationale space, and some decisions have no documented discussion in the archive, meaning that a correct match may not exist at all.

9. Conclusion and Future Work

We develop a framework that systematically generates rich training data, **RFCAlign**, by leveraging LLMs and the IETF mail archive. Models trained on this synthetic data outperform baselines on the downstream rationale-to-decision retrieval task over email discussions. For future work, we plan to extend the setting to open-archive retrieval. To make retrieval explanation-aware, we will reformulate design decisions in an explanation-oriented style that makes the rationale relationship explicit. Concretely, we will rewrite design decisions on a case-by-case basis following the rationale genres in Section 3.2, thereby indicating the kind of rationale the model is expected to retrieve. We will also examine how to incorporate text-generation features that allow LLMs to synthesize insights from email discussions and relevant RFC excerpts into clear and concise explanations.

Acknowledgments

We acknowledge the use of Microsoft Copilot for assistance with rephrasing, grammar checking, improving the readability of this manuscript, and contributing to the naming of our datasets. We would also like to thank Andrey Kutuzov, Étienne Simon, and Egil Rønningstad of the Language Technology Group (LTG) for their careful review of the draft.

Bibliographical References

Zhijie Bao, Wei Chen, Shengze Xiao, Kuang Ren, Jiaao Wu, Cheng Zhong, Jiajie Peng, Xuanjing Huang, and Zhongyu Wei. 2023. Disc-medllm: Bridging general large language models and real-world medical consultation. *arXiv preprint arXiv:2308.14346*.

Jie Bian, Nikolay Arefev, Max Mühlhäuser, and Michael Welzl. 2025. Automated insights into github collaboration dynamics. *IEEE Access*, 13:85526–85542.

Jie Bian, Michael Welzl, Andrey Kutuzov, and Nikolay Arefyev. 2024. Tell me why: Language models help explain the rationale behind internet protocol design. In *2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, pages 447–453. IEEE.

Scott Bradner. 1998. Ietf working group guidelines and procedures. Technical report.

Yaping Chai, Haoran Xie, and Joe S Qin. 2025. Text data augmentation for large language models: A comprehensive survey of methods, challenges, and opportunities. *arXiv preprint arXiv:2501.18845*.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*.

Alissa Cooper and Paul E. Hoffman. 2020. Working Group GitHub Administration. RFC 8875.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, et al. 2025. Mmteb: Massive multilingual text embedding benchmark. *arXiv preprint arXiv:2502.13595*.

A Farrel. 2014. Rfc 7221: Handling of internet-drafts by ietf working groups.

SU Hongjin, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, et al. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*.

Quzhe Huang, Mingxu Tao, Chen Zhang, Zhenwei An, Cong Jiang, Zhibin Chen, Zirui Wu, and Yansong Feng. 2023. Lawyer llama technical report. *arXiv preprint arXiv:2305.15062*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Ishan Jindal, Chandana Badrinath, Pranjal Bharti, Lakkidi Vinay, and Sachin Dev Sharma. 2024. Balancing continuous pre-training and instruction fine-tuning: Optimizing instruction-following in llms. *arXiv preprint arXiv:2410.10739*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Prashant Khare, Mladen Karan, Stephen McQuistin, Colin Perkins, Gareth Tyson, Matthew Purver, Patrick Healey, and Ignacio Castro. 2022. The web we weave: Untangling the social graph of the ietf. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 500–511.

Warren Kumari and Erik Kline. 2020. Rfc 8910: Captive-portal identification in dhcp and router advertisements (ras).

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Nicholas Lee, Thanakul Wattanawong, Sehoon Kim, Karttikeya Mangalam, Sheng Shen, Gopala Anumanchipalli, Michael Mahoney, Kurt Keutzer, and Amir Gholami. 2024. Llm2llm: Boosting llms with novel iterative data enhancement. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 6498–6526.

Matthew Lepinski and Kotikalapudi Sriram. 2017. Rfc 8205: Bgpsec protocol specification.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Sihang Li, Jin Huang, Jiaxi Zhuang, Yaorui Shi, Xiaochen Cai, Mingjun Xu, Xiang Wang, Linfeng Zhang, Guolin Ke, and Hengxing Cai. Scilitllm: How to adapt llms for scientific literature understanding. In *The Thirteenth International Conference on Learning Representations*.

Jeng-Wei Lin, Yi-Ting Lin, and Fang-Yie Leu. 2023. Case study in generating rfc maps of ietf standards. In *International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 228–237. Springer.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2732–2747.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

Stephen McQuistin, Mladen Karan, Prashant Khare, Colin Perkins, Gareth Tyson, Matthew Purver, Patrick Healey, Waleed Iqbal, Junaid Qadir, and Ignacio Castro. 2021. Characterising the ietf through the lens of rfc deployment. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 137–149.

Roberto Morabito and Jaime Jiménez. 2020. Ietf protocol suite for the internet of things: Overview and recent advancements. *IEEE Communications Standards Magazine*, 4(2):41–49.

Chris Morrow, Warren Kumari, Rob Austein, Steven Bellovin, Russ Housley, Stephen Kent, Matt Lepinski, and Kotikalapudi Sriram. 2018. Rfc 8374-bgpsec design choices.

Niklas Muennighoff, SU Hongjin, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. In *ICLR 2024 Workshop: How Far Are We From AGI*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Mihai Nadăș, Laura Dioșan, and Andreea Tomescu. 2025. Synthetic data generation using large language models: Advances in text and code. *IEEE Access*.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826.

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

Siamak Shakeri, Cicero dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. Reasonir: Training retrievers for reasoning tasks. *arXiv preprint arXiv:2504.20595*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

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

Orion Weller, Benjamin Van Durme, Dawn Lawrie, Ashwin Paranjape, Yuhao Zhang, and Jack Hessel. 2024. Promptriever: Instruction-trained retrievers can be prompted like language models. *arXiv preprint arXiv:2409.11136*.

Michael Welzl, Stephan Oepen, Cezary Jaskula, Carsten Griwodz, and Safiqul Islam. 2021. Collaboration in the ietf: an initial analysis of two decades in email discussions.

Ran Xu, Hejie Cui, Yue Yu, Xuan Kan, Wenqi Shi, Yuchen Zhuang, May Dongmei Wang, Wei Jin, Joyce Ho, and Carl Yang. 2024. Knowledge-infused prompting: Assessing and advancing clinical text data generation with large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 15496–15523.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. 2025. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. *arXiv preprint arXiv:2503.02951*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Xuanjing Huang, et al. 2023. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *CoRR*.

Appendix

A. Prompt

The instruction shown in Figure 2 is used and referred to as the *Judge–Then–Infer* strategy. The model must return a JSON object with two fields: `relevance` $\in$ {Yes, No} and `snippet` (the synthesized decision). We use `relevance` to filter generations: entries labeled `No` are excluded from the dataset.

B. Analyzed Example

We present the examples of our analyzed error cases in the Figures 3 and 4 below. For RFC 8335, we provide the relevant email,[10] with real URLs literally replaced by "URL," greetings and personal identifiers removed, and the quote symbol (>) stripped out. The email is typical of the archive, with quotes from earlier thread messages and occasional excerpts from related Internet-Drafts or RFCs, relevant or otherwise.

C. Full Results

Tables 4 and 5 report the results obtained from training the model on all four synthetic data types.

[10] https://mailarchive.ietf.org/arch/msg/int-area/I8ihR82CRqGEyWAY00GmmHKKjDY/

> Your task is to analyze the provided IETF email to determine its relevance to technical discussions surrounding RFCs or Internet-Drafts (I-Ds). Please follow these guidelines:
> 1. Classification: Assess the email to determine if it should be processed. Ignore the email if it is primarily a social communication, routine announcement, or automatically generated notification (e.g., notices about new versions of I-Ds). Focus on the email if it contains technical discussions, debates, or design decisions related to network protocols, standards, or associated themes.
> 2. Relevance Assessment: If the email is deemed relevant (i.e., technical discussion), proceed to generate the snippet. If it is ruled out as irrelevant, skip to the next email.
> 3. Snippet Generation:
> If the email directly cites or quotes text from an RFC or I-D, use that quoted text as is.
> If there are no direct quotes, synthesize a plausible text snippet that reflects the formal grammar, terminology, and style typically found in RFCs or I-Ds based on the email content.

Figure 2: Prompt demonstration

GitHub	BM25	P	M^R	M^S variants			
				QN	QV	LN	LV
Test	63.23	67.76	**79.27**	74.00	72.11	70.77	70.83
OOD	41.49	46.71	**57.16**	50.27	47.61	48.22	46.85

Table 4: MRR@10 results. Abbreviations: M (Mistral), R (real-world GitHub), S (synthetic), Q (Qwen), L (Llama), N (non-verbose), V (verbose), P (Promptriever). Bold = highest; underline = second-highest.

Email	BM25	P	M^R	M^S variants			
				QN	QV	LN	LV
RFC 7657	54.33	60.60	60.30	61.97	**63.49**	58.13	61.40
RFC 8205	73.91	70.83	81.96	**82.50**	82.14	78.99	79.31
RFC 8335	56.74	46.41	46.52	**67.74**	52.48	63.78	60.52
RFCs	60.35	59.23	62.22	**68.11**	65.09	64.06	65.14

Table 5: MRR@5 results. Abbreviations: M (Mistral), R (real-world GitHub), S (synthetic), Q (Qwen), L (Llama), N (non-verbose), V (verbose), P (Promptriever). Bold = highest; underline = second-highest.

D. Emails

In the table 6 below, we summarize the statistics of the emails used in our pipeline and the corresponding synthetic outputs generated by the models. Not all collected emails are included in the final dataset, as some fail to produce valid generations even when submitted correctly to the prompting stage. For instance, in the ACE working group, we collected 5 699 emails. When these emails were combined with the prompt and processed by the LLM to generate synthetic decisions, Qwen pro-

Query (Rationale):
Possibly, an off-router box (one or more per AS) can be used that performs path validation. For example, these capabilities might be incorporated into a route reflector. At an **ingress** router, one needs the Adj-RIB-In entries validated but not the RIB-out entries. So, the off-router box is probably unlike the traditional route reflector; it sits at the network **edge** and validates all incoming BGPsec updates. Thus, it appears that each router passes each BGPsec update it receives to the off-router box and receives a validation result before it stores the route in the Adj-RIB-In. Question: What about failure modes here? The failure modes would be dependent on the following: How much of the control plane is outsourced. How reliable the off-router box is (or, equivalently, communication to and from it). How centralized vs. distributed this arrangement is. When any kind of outsourcing is done, the user needs to be watchful and ensure that the outsourcing does not cross trust/security boundaries.

Document (Decision 1 (gold)):
Update signature validation and signing can be outsourced to an off-board server or processor.

Retrieved Top 1 (Decision 2) by Mistral^R:
BGPsec validation need be performed only at eBGP **edge**s. The validation status of a BGP signed/unsigned update may be conveyed via iBGP from an **ingress edge** router to an egress **edge** router. Local policy in the AS will determine how the validation status is conveyed internally, using various preexisting mechanisms, e.g., setting a BGP community, or modifying a metric value such as Local_Pref or MED. A signed update that cannot be validated (except those with syntax errors) should be forwarded with signatures from the **ingress** router to the egress router, where it is signed when propagated towards other eBGPsec speakers in neighboring ASes. Based entirely on local policy settings, an egress router may trust the validation status conveyed by an **ingress** router, or it may perform its own validation. The latter approach may be used at an operator's discretion, under circumstances when RPKI skew is known to happen at different routers within an AS. An extended community for carrying the origin validation state in iBGP has been specified in RFC 8097 [RFC8097].

Figure 3: Case study of the RFC 8205; Mistral^S fails to retrieve the gold decision among top 10.

Query (Rationale):
All codes a First Come First Serve (FCFS). I will spin a new version and submit it ASAP. We need to ask for one more thing that we didn't ask for in our review, and it's because there's an error in the registration procedure data in the registries. Both URL and URL incorrectly state that the IESG Approval or Standards Action are the registration procedures for all "Code" registries. However, RFC 4443 and RFC 2780 state that "The policy for assigning Code values for new IPv6 ICMP Types not defined in this document should be defined in the document defining the new Type". URL and "The policy for assigning Code values for new IPv4 ICMP Types should be defined in the document defining the new Type value" URL. Can you provide registration procedures for all of the Code registries? For our purposes, these could be added to the document as late as after IESG approval, when we're completing the actions, if that's more convenient/appropriate. Was our identification of 15 as the maximum values for the Code registries correct? I have posted a new version of draft-ietf-intarea-probe reflecting IETF LC comments. Please take a look at the document to make sure that I have addressed your comments satisfactorily. Abstract: This document describes a network diagnostic tool called PROBE. PROBE is similar to PING, in that it can be used to query the status of a probed interface. It differs from PING in that it does not require bidirectional connectivity between the probing and probed interfaces. Instead, PROBE requires bidirectional connectivity between the probing interface and a proxy interface. The proxy interface can reside on the same node as the probed interface or it can reside on a node to which the probed interface is directly connected. This document updates RFC 4884. Please note that it may take a couple of minutes from the time of submission until the htmlized version and diff are available at tools.ietf.org.

Document (Decision 1 (gold)):
All codes mentioned above are assigned on a First Come First Serve (FCFS) basis with a range of 0 -255.

Retrieved Top 1 (Decision 2) by Mistral^R:
This document describes a network diagnostic tool called PROBE. PROBE is similar to PING, in that it can be used to test the status of a probed interface. It differs from PING in that it does not require bidirectional connectivity between the probing and probed interfaces. Alternatively, PROBE requires bidirectional connectivity between the probing interface and a proxy interface. The proxy interface can reside on the same node as the probed interface or it can reside on a node to which the probed interface is directly connected. This document updates RFC 4884.

Figure 4: Case study of the RFC 8335; Mistral^S fails to retrieve the gold decision among top 10. The same light–dark shading denotes overlapping content.

duced 5 532 outputs and Llama produced 5 632. In both cases, the number of generations is lower than the number of input emails. This behavior is expected in large-scale synthetic data pipelines: some inputs inevitably fail to produce outputs due to operational factors rather than problems with the prompt or the data itself. Typical causes include inference timeouts, transient service disruptions, rate-limit handling, backend execution failures, or retry attempts that eventually terminate. These practical issues naturally create a gap between the number of available emails and the number of successfully generated synthetic decisions.

NO.	WG	Emails	Qwen	Llama	NO.	WG	Emails	Qwen	Llama
1	ace	5 699	5 532	5 632	33	netconf	20 349	20 289	20 194
2	acme	4 987	4 980	4 963	34	nmop	1 508	1 508	1 498
3	add	4 535	4 451	4 496	35	nwcrg	1 418	1 418	1 407
4	alto	6 189	6 181	6 170	36	oauth	25 975	25 916	25 896
5	anima	7 300	7 287	7 275	37	ohai	446	446	434
6	avt	19 296	19 281	19 247	38	opsawg	11 379	11 352	11 301
7	captive-portals	1 258	1 256	1 254	39	pearg	666	666	647
8	ccamp	23 404	23 346	23 327	40	perc	1 079	1 077	1 047
9	ccwg	447	447	443	41	ppm	601	599	583
10	cellar	3 460	3 455	3 440	42	privacy-pass	700	700	682
11	core	13 916	13 799	13 899	43	quic	11 201	11 182	11 139
12	cose	5 308	5 298	5 269	44	rats	4 875	4 871	4 840
13	detnet	5 616	5 596	5 566	45	roll	12 966	12 945	12 897
14	dhcwg	19 894	19 885	19 860	46	rtcweb	18 152	18 141	18 072
15	dmarc	12 626	12 604	12 594	47	sacm	6 416	6 412	6 406
16	dnsop	32 758	32 724	32 718	48	scitt	2 518	2 510	2 508
17	dnssd	2 392	2 391	2 379	49	sedate	408	407	385
18	doh	1 829	1 825	1 802	50	sframe	279	278	270
19	emu	4 708	4 703	4 669	51	snac	909	909	884
20	homenet	8 609	8 597	8 590	52	spasm	12 807	12 781	12 765
21	httpapi	1 119	1 118	1 081	53	suit	2 173	2 171	2 156
22	ice	1 648	1 646	1 629	54	t2trg	1 469	1 466	1 422
23	ietf-and-github	912	906	889	55	taps	2 686	2 683	2 678
24	ietf-http-wg	41 296	41 151	41 194	56	tls	34 637	34 540	34 530
25	iot-onboarding	311	311	306	57	tm-rid	3 089	3 086	3 055
26	jsonpath	1 159	1 151	1 105	58	txauth	2 511	2 507	2 499
27	lake	1 133	1 131	1 092	59	webpush	927	927	906
28	masque	1 657	1 655	1 601	60	webtransport	582	582	567
29	mls	2 040	2 033	2 021	61	wimse	908	908	902
30	mops	591	591	587	62	wish	854	853	835
31	moq	1 333	1 333	1 292	63	wpack	343	343	337
32	multipathtcp	3 959	3 952	3 932		**Total**	426 220	425 089	424 064

Table 6: Summary statistics of the synthetic data. *WG* represents the working group name, *Emails* the number of source emails used for generating each synthetic decision, and *Qwen* and *Llama* the LLMs used as generators. All emails are sourced from the IETF mailarchive, except the ietf-http-wg, which is archived at w3c.