Re^3: Relevance & Recency Retrieval for Mitigating Temporal Hallucination

Jiawei Cao[1], **Jie Ouyang**[1], **Mingyue Cheng**[1*], **Zhaomeng Zhou**[1]
Yupeng Li[1], **Zirui Liu**[1], **Chunli Liu**[3], **Shijin Wang**[1,2]
[1]University of Science and Technology of China, [2]iFLYTEK Research
[3]Hefei University of Technology, [*]Corresponding author
{cjwyv, ouyang_jie, zhoumh, liyupeng, liuzirui}@mail.ustc.edu.cn
mycheng@ustc.edu.cn liuchunli@hfut.edu.cn sjwang3@iflytek.com

Abstract

Retrieval-Augmented Generation (RAG) is a mainstream approach to mitigating hallucinations in Large Language Models (LLMs), yet in dynamic real-world scenarios, such as weather forecasting or evolving news events, existing retrievers suffer from both temporal-semantic misalignment and outdated-document interference. To address this, we propose **Re**levance & **Re**cency **Re**trieval (Re^3), a novel framework that mitigates temporal hallucinations via two core components: a Time-Aware Dual Relevance Encoder that embeds heterogeneous temporal signals into the semantic space to ensure retrieval fidelity, and a Conflict-Aware Recency Filter that performs listwise arbitration to identify and suppress obsolete factual versions. To rigorously evaluate this setting, we introduce Re^2 Bench, a large-scale benchmark comprising over 1.3 million instances designed to assess system robustness in realistic environments where temporal constraints and conflicting factual versions coexist. Experiments on three public benchmarks and Re^2 Bench demonstrate that Re^3 consistently outperforms the strongest baselines by an average of 9.7% in generation accuracy, with gains of up to 25.2% on challenging dynamic tasks, while demonstrating robustness across diverse RAG settings. Code and benchmark have been released to support reproducibility.[1]

1 Introduction

Retrieval-Augmented Generation (RAG) has emerged as a standard paradigm to mitigate hallucinations in Large Language Models (LLMs) by grounding generation in external evidence (Lewis et al., 2020; Bommasani et al., 2021; Guu et al., 2020; Liu et al., 2024). While effective for static knowledge, the reliability of RAG hinges on a critical assumption: retrieved evidence must be not only factually accurate but also temporally valid at the time of the query (Oche et al., 2025; Wu et al., 2024; Cheng et al., 2025; Liu et al., 2025).

In real-world dynamic environments, the assumption of temporal validity frequently breaks down. Following the broader, factuality-oriented definition of hallucinations in recent LLM literature (Huang et al., 2025; Yu et al., 2025), we distinguish temporal hallucination from standard generation-only fabrication (where a model simply invents unsupported claims). Instead, temporal hallucination occurs when an LLM generates answers that are factually incorrect or misleading regarding the current state of the world, driven by the retrieval of stale information. In such cases, the generator faithfully relies on retrieved evidence that is topically relevant but temporally obsolete, rather than the up-to-date ground truth.

Although recent work has explored incorporating time into RAG, existing methods either rely on heuristic or additive temporal biases (Gade et al., 2025; Abdallah et al., 2025) or treat time as a separate verification signal rather than an intrinsic semantic dimension (Zhang et al., 2025b), limiting their ability to model complex temporal-semantic interactions and resolve conflicting factual versions. We identify two fundamental retrieval-side challenges that drive these failures:

(i) Temporal-semantic misalignment. Temporal information in large-scale corpora is often heterogeneous and implicit (e.g., varying granularities or relative terms like "last year" without month or day). Existing retrievers that treat time as a rigid metadata filter are brittle under such ambiguity, leading to the under-retrieval of temporally relevant documents that lack precise timestamp matches.

(ii) Outdated document interference. Knowledge bases must preserve historical records to maintain complementary context (Xin et al., 2024), causing conflicting factual versions (e.g., different U.S. presidents across years) to coexist. Standard semantic matching cannot distinguish these obsolete

[1] Project page: https://github.com/cjwyv/Re3

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 25735–25760

versions from current ones solely based on topical overlap, often retrieving outdated but semantically rich documents that mislead the generator (Vu et al., 2024; Ouyang et al., 2025).

To address these challenges, we propose **Re**^{3}, a retrieval-side framework that disentangles and jointly models temporal relevance and recency. Re^3 is designed to be robust against the noise and sparsity of real-world temporal data. It integrates two complementary components that directly target the above challenges: a **Time-Aware Dual Relevance Encoder**, which injects heterogeneous temporal signals into the semantic space via feature-wise modulation (FiLM) and thus alleviates temporal-semantic misalignment; and a **Conflict-Aware Recency Filter**, which explicitly models factual versions and performs listwise arbitration over retrieved candidates so that the most recent consistent versions are promoted while outdated conflicting evidence is suppressed. To rigorously assess Re^3 under realistic temporal complexity, we further construct **Re**^{2} **Bench**, a large-scale evaluation benchmark where heterogeneous temporal expressions and conflicting factual versions are the norm.

Empirical evaluations on three public benchmarks and our proposed Re^2 Bench demonstrate that Re^3 achieves an average accuracy improvement of 9.7% over state-of-the-art baselines, with the most significant gains (up to 25.2%) on highly dynamic tasks like NOAA weather forecasting where conflicting factual versions are prevalent. Furthermore, sensitivity analysis confirms that our framework is model-agnostic, delivering consistent improvements regardless of the underlying retrieval backbone or generator architecture.

The contributions of this paper are as follows:

- We formalize the problem of temporal hallucination as a dual challenge of misalignment and interference.
- We propose **Re**^{3}, a retrieval-side framework that mitigates hallucinations in RAG, combining a Time-Aware Dual Relevance Encoder and a Conflict-Aware Recency Filter. Furthermore, we construct **Re**^{2} **Bench** with 1.3 million instances to rigorously evaluate future research under this setting.
- We demonstrate through extensive experiments that Re^3 sets a new state-of-the-art, outperforming strong baselines by an average of 9.7% in generation accuracy and exhibiting robustness across diverse tasks.

2 Related Work

2.1 Temporal Hallucination in RAG

While RAG effectively reduces general hallucinations by grounding LLMs in external evidence (Lewis et al., 2020; Guu et al., 2020; Lin et al., 2026; Liu et al., 2026), it introduces a new vulnerability: retrieval-induced hallucination. This occurs when the retriever returns documents that are misleading with respect to the query, and the generator uncritically incorporates these retrieved contents into its response (Yoran et al., 2024). In time-sensitive scenarios, this manifests as temporal hallucination, where LLMs generate obsolete responses by attending to semantically relevant but temporally invalid evidence (Chen et al., 2021).

Unlike temporal reasoning errors, which stem from the outdated internal weights of LLMs (Kasai et al., 2023), temporal hallucinations in RAG are primarily data-driven. The core challenge lies in the nature of real-world corpora: they are often non-stationary and unstructured, containing conflicting factual versions (e.g., different Presidents of the US) without explicit version control (Vu et al., 2024). Existing benchmarks (Chen et al., 2021; Wu et al., 2024) typically evaluate temporal QA in controlled environments with clean timestamps. However, they overlook the poisoning effect of outdated documents in realistic settings where conflicting versions coexist. Our work bridges this gap by focusing on the detection and mitigation of such outdated evidence during the retrieval stage.

2.2 Time-Sensitive Retrieval

Traditional approaches to time-sensitive retrieval rely heavily on lexical matching combined with metadata filtering or decay functions. Early works integrated publication dates into ranking functions (e.g., recency priors) to favor newer documents (Li and Croft, 2003; Zhang et al., 2025a). In the neural era, methods have evolved to incorporate timestamps as distinct modalities. For instance, previous works have proposed creating time-aware embeddings by concatenating timestamp vectors with semantic embeddings (Rosin et al., 2017; Gade et al., 2025; Abdallah et al., 2025; Zhang et al., 2025b) or using dedicated temporal knowledge graph embeddings (Goel et al., 2019).

However, applying these methods to RAG faces

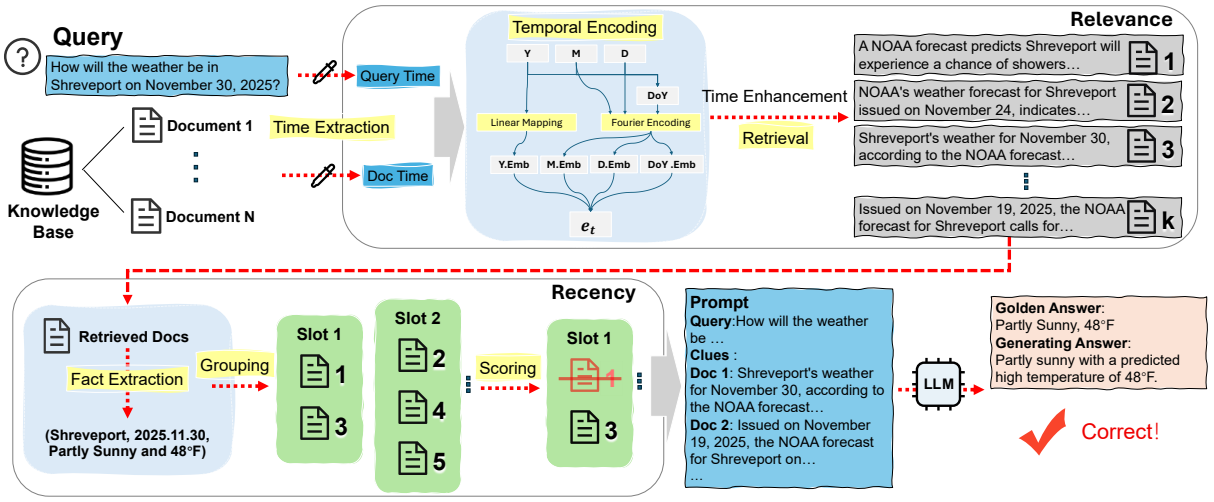

Figure 1: Overview of Re^3 (Relevance & Recency Retrieval). Given a time-sensitive query, Re^3 first extracts temporal signals from both the query and candidate documents, and encodes timestamps into temporal embeddings that modulate semantic representations in the Time-Aware Dual Relevance Encoder. The retrieved documents are then refined by a Conflict-Aware Recency Filter, which groups extracted facts into slots, identifies the most recent factual versions based on publication time, and filters outdated evidence before generation. After recency refinement, the remaining evidence enables the RAG system to produce temporally correct answers.

two significant hurdles. First, most existing time-aware retrievers assume the availability of structured, complete timestamps (e.g., YYYY-MM-DD) for all documents. They struggle with the heterogeneity of real-world data, where temporal signals may be implicit, coarse-grained (year only), or missing entirely (Gao et al., 2024). Second, current reranking strategies often treat recency as a global heuristic (newer is better), failing to distinguish between timeless facts (which do not expire) and transient facts (which become outdated). Recent works like MRAG (Zhang et al., 2025b) attempt to handle multi-aspect retrieval but lack specific mechanisms for fact-level version control. In contrast, our Re^3 framework explicitly models time as a continuous semantic feature to handle heterogeneous signals and employs a conflict-aware mechanism to filter outdated versions specifically.

3 Re^3: Relevance & Recency Retrieval

As shown in Figure 1, Re^3 tackles temporal hallucination in RAG through a retrieval-side framework with relevance and recency modeling. The relevance module incorporates temporal signals to enhance semantic embeddings, while the recency module filters outdated documents by reasoning over conflicting versions.

3.1 Preliminaries

We introduce the basic notation and retrieval setting used throughout this work.

Data Representation. Let q denote a user query and d a document. A query q consists of query content q_c and an associated query time q_t, which may be partially missing. Each document d contains document content d_c, a document time d_t mentioned in or associated with the content, and a publication time d_{pub}. In real-world corpora, d_t may be incomplete or heterogeneous, while d_{pub} is always available and provides a reliable chronological order among document versions.

Retrieval and Generation. A retriever is designed to encode the contents of the query and documents using a shared encoder $\text{Enc}(\cdot)$:

$$\mathbf{q} = \text{Enc}(q_c), \qquad \mathbf{d} = \text{Enc}(d_c), \tag{1}$$

and computes semantic relevance via inner product

$$s_{\text{sem}}(q, d) = \langle \mathbf{q}, \mathbf{d} \rangle. \tag{2}$$

Documents are ranked by s_{sem}, and the top-k results are retrieved.

Given a query q and its retrieved documents $\{d_1, \ldots, d_k\}$, a generator $\text{Gen}(\cdot)$ produces an answer by conditioning on them:

$$a = \text{Gen}\big(\text{Prompt}(q_c, d_{c_1}, \ldots, d_{c_k})\big), \tag{3}$$

where a denotes the generated output. As a result, retrieval quality directly determines the evidence available to the generator, and temporal hallucinations can be induced by incorrect evidence.

Assumptions on Temporal Ground Truth. Our framework operates under the well-defined assumption that the underlying corpus consists of authoritative, time-series sources where the chronologically latest record for a given entity or event represents the current ground truth. Our primary objective is to prevent outdated, albeit once-correct, records from misleading the generator. Consequently, scenarios where the most recent document itself contains false or adversarial information are intentionally considered out of scope. Addressing misinformation, multi-source conflicts, or source unreliability requires orthogonal techniques, such as cross-source contradiction detection and source-reliability estimation. We consider these credibility challenges complementary to our focus on resolving temporal outdatedness and further discuss these boundaries in our Limitations section.

3.2 Temporal Parsing and Normalization

To bridge the gap between queries and time-sensitive retrieval, this module transforms diverse temporal mentions—ranging from precise dates to coarse-grained eras—into actionable constraints.

Let x denote an input text sequence. Our goal is to parse x into a standardized tuple $\mathcal{T} = (y, m, dom, \mathbf{v})$, where y, m, dom represent the year, month, and day-of-month, and $\mathbf{v} = (v_y, v_m, v_{dom}) \in \{0,1\}^3$ is a binary mask vector indicating the availability of each component. Unlike rigid rule-based parsers, we employ a Transformer-based temporal extractor to handle implicit or multi-granular expressions. Specifically, we encode x using a pre-trained encoder (e.g., BERT) to obtain the global representation $\mathbf{h}_{\text{CLS}}$. This embedding is then fed into parallel linear classification heads to predict the probability distributions for year, month, and day-of-month independently. For coarse-grained inputs like "the 1990s", the model predicts a representative year (e.g., $y = 1995$) while setting the month/day-of-month masks to zero ($v_m = v_{dom} = 0$). Similarly, "May 2024" is normalized to $(y = 2024, m = 5)$ with $v_{dom} = 0$.

3.3 Time-Aware Dual Relevance Encoder

Standard dense retrievers often ignore temporal discrepancies during embedding. To overcome this, we propose a time-aware encoder that fuses continuous temporal signals into the semantic representation via a modulation mechanism.

Continuous Time Representation. Given the parsed temporal tuple $\mathcal{T} = (y, m, dom, \mathbf{v})$, we construct a continuous time embedding $\mathbf{e}_{\text{time}} \in \mathbb{R}^D$. We compute doy only when all three components (y, m, dom) are available. Specifically, we independently encode four temporal components (year, month, day-of-month, day-of-year), projecting each into a sub-space of dimension $D/4$.

For the non-cyclic year y, we employ a normalized linear projection via an MLP:

$$\mathbf{h}_y = \text{MLP}_y\left(\frac{y - \mu_y}{\sigma_y}\right) \odot v_y \tag{4}$$

where μ_y, σ_y are corpus statistics, and $v_y \in \{0,1\}$ is the corresponding mask bit from $\mathbf{v}$. Similarly, dom is mapped to a dense vector $\mathbf{h}_{dom}$ via a learnable embedding table to capture discrete irregularities, masked by v_{dom}.

For cyclic components (month m and day-of-year doy), we utilize projected sinusoidal embeddings to capture periodicity. Unlike standard multi-frequency positional encodings, we explicitly model the fundamental period T of each component. Specifically, we first extract the fundamental cyclic features for a scalar input x:

$$\boldsymbol{\phi}(x, T) = \left[\sin\left(\frac{2\pi x}{T}\right), \cos\left(\frac{2\pi x}{T}\right)\right] \in \mathbb{R}^2. \tag{5}$$

We then obtain the embeddings for month ($T = 12$) and day-of-year ($T = 366$) by projecting these features through learnable linear transformations:

$$\mathbf{h}_m = \boldsymbol{\phi}(m, 12)\, \mathbf{W}_m \odot v_m, \tag{6}$$

$$\mathbf{h}_{doy} = \boldsymbol{\phi}(doy, 366)\, \mathbf{W}_{doy} \odot \text{Exist}_{doy}. \tag{7}$$

Here, $\mathbf{W}_m, \mathbf{W}_{doy} \in \mathbb{R}^{2\times(D/4)}$ are learnable projection matrices mapping the 2D cyclic signals into the embedding space. Note that for the implicit day-of-year component, the mask Exist_{doy} is active (1) only if the full date y, m, dom is available.

Finally, we concatenate these four vectors to form $\mathbf{e}_{\text{time}} = [\mathbf{h}_y; \mathbf{h}_m; \mathbf{h}_{dom}; \mathbf{h}_{doy}]$. Crucially, unobserved components are embedded to zeros, ensuring that the temporal modulation remains robust to missing values and adapts to the variable granularity of real-world queries.

Temporal-Semantic Fusion via FiLM. Let $\mathbf{e}_{\text{sem}} \in \mathbb{R}^D$ be the semantic embedding from a pre-trained transformer. We inject temporal information using Feature-wise Linear Modulation (FiLM) (Perez et al., 2018). Unlike simple concatenation, FiLM allows the time signal to scale and shift specific semantic dimensions:

$$\gamma, \beta = \text{MLP}_{\text{proj}}(\mathbf{e}_{\text{time}}), \tag{8}$$

$$\mathbf{e}_{\text{final}} = (1 + \alpha\gamma) \odot \mathbf{e}_{\text{sem}} + \beta. \tag{9}$$

Here, $\gamma, \beta \in \mathbb{R}^D$ are the scaling and shifting coefficients derived from time, and α is a learnable scalar controlling the strength of temporal modulation. Intuitively, the scaling factor γ highlights dimensions sensitive to time (e.g., "president"), while the shift β re-orients the representation towards the specific era (e.g., "2020s context"). The term $(1 + \dots)$ creates a residual-like structure, ensuring that the model retains strong semantic capabilities even when temporal signals are weak. We ℓ_2-normalize the final embeddings.

Training Objective. We train the dual encoder with a joint objective that combines semantic discrimination with temporal robustness. The primary component is a symmetric InfoNCE loss (van den Oord et al., 2019; Zhou et al., 2025), which aligns each query q with its positive document d^+ against in-batch negatives:

$$\mathcal{L}_{\text{nce}} = \frac{1}{2}\Big(\mathcal{L}_{q\to d} + \mathcal{L}_{d\to q}\Big), \tag{10}$$

where $\mathcal{L}_{q\to d}$ is the cross-entropy loss over rows of the similarity matrix $s(q_i, d_j) = \langle \mathbf{e}_i^{(q)}, \mathbf{e}_j^{(d)} \rangle / \tau$ with the diagonal as positives, and $\mathcal{L}_{d\to q}$ is defined analogously over columns.

To encourage temporally consistent rankings, we add a margin-based regularizer that enforces the positive pair to score higher than the hardest in-batch negative. Let B denote the batch size. We minimize the average margin loss over the batch:

$$\mathcal{L}_{\text{hard}} = \frac{1}{B}\sum_{i=1}^{B} \max\big(0,\ \delta + s_i^- - s_i^+\big), \tag{11}$$

where s_i^- denotes the maximum similarity between q_i and any standard in-batch negative (i.e., positive documents paired with other queries in the same batch), and s_i^+ is the similarity of the matched query–document pair. δ is a margin hyperparameter enforcing a minimum separation.

Then we add a separate margin loss for explicitly retrieved time-wrong negatives. Here, d^+ denotes the positive document paired with query q, and d_{tw}^- denotes the time-wrong negative document specifically retrieved for the same query (disjoint from standard in-batch negatives):

$$\mathcal{L}_{\text{tw}} = \frac{1}{B}\sum_{i=1}^{B} \max\big(0,\ \delta + \langle q_i, d_{\text{tw},i}^- \rangle - \langle q_i, d_i^+ \rangle\big). \tag{12}$$

The final objective is a weighted sum, where λ_{hard} and λ_{tw} control the contributions of the two margin-based regularizers:

$$\mathcal{L} = \mathcal{L}_{\text{nce}} + \lambda_{\text{hard}}\mathcal{L}_{\text{hard}} + \lambda_{\text{tw}}\mathcal{L}_{\text{tw}}. \tag{13}$$

3.4 Conflict-Aware Recency Filter

While the Time-Aware Encoder improves retrieval relevance, it cannot filter out outdated documents. As formally defined in preliminaries, our framework assumes that within credible corpora, the chronologically latest version of conflicting facts supersedes prior ones. We introduce a Recency Filter that explicitly models factual versions and suppresses outdated information within the retrieved candidates.

Efficient Listwise Fact Extraction. Given the query q and the initial top-k candidates $\mathcal{D}_k$ retrieved by the encoder, we concatenate their contents and feed the resulting sequence into a lightweight LLM (e.g., Llama-3-8B) and then extract relevant factual triples (h, r, v) in a single inference pass. Our listwise approach processes all k candidates in a single LLM forward pass. Moreover, processing documents jointly allows the LLM to implicitly perform entity resolution (e.g., aligning "Biden" with "Joe Biden"), ensuring consistent version comparison.

Conflict-Based Filtering. With the extracted facts grouped by relation (h, r), we identify the most current version within each group by d_{pub}. Then we refine the candidate list by filtering outdated documents. For a conflict cluster, we designate the version v^* associated with the latest publication timestamp as the winning version.

Based on this ground truth, we compute a recency vote $s_{\text{rec}}(d)$ for each document d:

$$s_{\text{rec}}(d) = \begin{cases} +1, & \text{if } d \text{ supports } v^*, \\ -1, & \text{if } d \text{ supports } v' \neq v^*, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

	TimeQA			Nobel-TSRAG			HoH		
Model	R@5	MRR	Acc	R@5	MRR	Acc	R@5	MRR	Acc
BM25	0.3843	0.2725	0.1996	0.5720	0.3987	0.4049	0.5825	0.4698	0.4887
BGE-M3	0.8290	0.5912	0.7272	0.8718	0.6618	0.7879	0.7900	0.5600	0.5900
BGE-reranker	0.8370	0.6224	0.7314	0.8764	0.6644	0.7932	**0.8438**	**0.6178**	0.6025
TempRALM	0.8403	0.6006	0.7106	0.8348	0.6023	0.7579	0.7900	0.5600	0.5900
TempRetrieval	0.8728	0.6117	0.7360	0.8741	0.6641	0.7811	0.7900	0.5600	0.5900
MRAG	0.8448	0.6014	0.7001	0.8415	0.6128	0.7684	0.7900	0.5600	0.5900
LoRA	0.6886	0.4618	0.6943	0.9000	0.7171	0.8299	0.7537	0.5226	0.5775
Re^3	**0.9280**	**0.6891**	**0.7743**	**0.9453**	**0.7243**	**0.9092**	0.7900	0.5600	**0.6339**

Table 1: Main results on public benchmarks. We report Recall@5 (R@5), Mean Reciprocal Rank (MRR), and Accuracy(Acc). On HoH, queries contain no temporal signals, causing all temporal retrievers to degenerate to their corresponding base retrievers as their retrieval metrics (R@5, MRR) are identical, while the recency-aware component of Re^3 remains effective and yields non-trivial Accuracy. Best results are highlighted in bold.

	Re^2 Bench–NYC			Re^2 Bench–COVID			Re^2 Bench–NOAA		
Model	R@5	MRR	Acc	R@5	MRR	Acc	R@5	MRR	Acc
BM25	0.2100	0.1422	0.1925	0.4812	0.3111	0.4138	0.3938	0.2568	0.3387
BGE-M3	0.6087	0.4524	0.5825	0.8775	0.5658	0.8438	0.5625	0.3598	0.4977
BGE-reranker	0.7850	0.6972	0.7837	0.8588	0.6034	0.8313	0.7550	0.3942	0.6625
TempRALM	0.8712	0.7678	0.8263	0.8812	0.6800	0.8370	0.6262	0.3946	0.3625
TempRetrieval	0.8650	0.8091	0.8137	0.8762	0.6489	0.8012	0.7300	0.4490	0.4238
MRAG	0.8662	0.7664	0.8263	0.8920	0.6803	0.8062	0.6350	0.3973	0.3575
LoRA	0.8937	0.6350	0.7800	0.8611	0.6354	0.8520	0.8925	0.5167	0.6975
Re^3	**0.9248**	**0.8906**	**0.8788**	**0.9525**	**0.7215**	**0.9112**	**0.9625**	**0.5929**	**0.8732**

Table 2: Main results on Re^2 Bench, which evaluates the impact of retrievers on mitigating temporal hallucination in RAG. We report Recall@5 (R@5), Mean Reciprocal Rank (MRR), and Accuracy(Acc). Best results are highlighted in bold.

We then apply a threshold-based filtering rule: documents with $s_{\text{rec}}(d) < 0$ are removed from the candidate set, ensuring that only temporally consistent evidence reaches the generator.

4 Experiments

4.1 Experimental Settings

Re^2 Bench. Standard benchmarks often lack the complexity required to evaluate temporal hallucinations under realistic conditions. To rigorously assess retrieval robustness, we construct Re^2 Bench, a large-scale benchmark comprising 1.3M instances from three time-sensitive domains: NYC Motor Vehicle Collisions (New York City Open Data, 2025), COVID-19 statistics (Mathieu et al., 2021), and NOAA weather forecasts (National Oceanic and Atmospheric Administration, 2025).

Domain	Instances	Primary Challenge
NYC	500K	Relevance
COVID	430K	Recency
NOAA	360K	Relevance & Recency

Table 3: Re^2 Bench statistics across three domains.

The construction consists of three key stages:

- **Data Cleaning & Normalization**: Raw records are parsed into structured tuples. To mirror real-world temporal heterogeneity, we randomly mask explicit dates in a subset of documents, forcing models to rely on coarse-grained or implicit temporal cues.

- **Negative Construction via Tuple Perturbation**: To facilitate contrastive learning and evaluate retrieval robustness, we systematically perturb specific attributes of the source tuples to generate highly realistic hard negatives. Specifically, we create two primary types of negatives: (i) *Time-wrong* negatives,

generated by shifting the timestamp while maintaining spatial and semantic attributes; and (ii) *Entity/Attribute-wrong* negatives, generated by altering non-temporal dimensions (e.g., location or status) while keeping the time constant. An illustrative example for the NOAA domain is shown in Table 4. These controlled perturbations yield hard negatives that maintain high topical similarity with the query but are factually incorrect, thereby rigorously testing the model's temporal sensitivity.

- **LLM-Driven QA Synthesis**: We leverage Large Language Models (LLMs) to generate diverse natural language queries and documents conditioned on these structured tuples (including the positive answer and the perturbed negatives). This ensures the resulting benchmark contains rich linguistic variation while strictly preserving the controlled factual and temporal constraints.

Candidate Type	Tuple Representation
Positive Answer	(2025-11-20, NYC, *rainy*, 12°C)
Time-wrong	(**2025-11-19**, NYC, *rainy*, 9°C)
Location-wrong	(2025-11-20, **Boston**, *sunny*, 12°C)

Table 4: An illustrative example of candidate construction via tuple perturbation. The perturbed attributes are highlighted in bold. These tuples are subsequently converted into natural language documents.

The final benchmark statistics are summarized in Table 3. NYC and COVID primarily stress-test temporal–semantic misalignment through heterogeneous timestamps, while NOAA additionally introduces outdated documents for the same query. As a result, NOAA yields dense clusters of conflicting factual versions. The detailed construction process is presented in Appendix A.

Public datasets. We additionally evaluate on three established datasets: (1) **TimeQA** (Chen et al., 2021), which evaluates temporal reasoning over explicit time expressions; (2) **Nobel-TSRAG** (Wu et al., 2024), derived from Nobel Prize archives with explicit year constraints; and (3) **HoH** (Ouyang et al., 2025), which focuses on outdated documents detection without explicit query-side temporal signals.

Baselines & Metrics. BM25 (Robertson and Zaragoza, 2009) serves as a traditional lexical baseline. For dense retrieval, we adopt vanilla BGE-M3 as the base encoder, and evaluate several time-aware extensions, including TempRALM (Gade et al., 2025) (temporal bias term), TempRetrieval (Abdallah et al., 2025) (temporal embedding concatenation), MRAG (Zhang et al., 2025b) (temporal constraint decomposition), and a LoRA-based dense retriever (Hu et al., 2022) with temporal supervision. BGE-reranker is used as a cross-encoder reranking baseline.

For retrieval performance, we report Recall@5 (R@5) and Mean Reciprocal Rank (MRR). For end-to-end QA evaluation, we use accuracy as the primary metric, measuring whether the generated answer exactly matches the gold answer, following common practice in RAG systems (Chen et al., 2021; Wu et al., 2024).

RAG Configuration. We adopt a unified RAG setup. We use BGE-M3 (Li et al., 2023; Chen et al., 2024) as the shared backbone for all time-aware dense retriever baselines. We retrieve the top-5 documents for answer generation, as prior studies show that large language models primarily rely on the top-ranked few passages (Reichman and Heck, 2024; Zhao et al., 2024). Across all RAG-based baselines and Re^3, we fix the generator to DeepSeek-V3.2 (DeepSeek-AI et al., 2025). To ensure that d_{pub} is visible to all models, we explicitly annotate each document with its d_{pub} (if available) in the prompt, and instruct the model to prioritize the most recent documents.

4.2 Main Results

Table 1 and Table 2 summarize the performance of Re^3 against state-of-the-art baselines across six diverse datasets. We also provide a detailed case study in Appendix B. Overall, Re^3 establishes a new state-of-the-art, achieving the highest accuracy and Recall@5 in 5 out of 6 benchmarks. We structure our analysis into three key observations:

Performance on Public Temporal Datasets. On standard benchmarks with explicit timestamps (TimeQA and Nobel-TSRAG), Re^3 demonstrates substantial improvements over existing temporal retrievers. Compared to TempRetrieval and MRAG, which rely on separate temporal scoring or additive biases, Re^3 improves Recall@5 by 5.5% on TimeQA and 7.1% on Nobel-TSRAG. This indicates that our Time-Aware Dual Relevance Encoder, which fuses temporal signals via deep modulation, captures temporal-semantic interactions more effectively than loose coupling strategies.

Adaptability to Implicit Temporal Signals. On the HoH dataset, queries typically lack explicit temporal constraints, serving as a stress test for model adaptability. Most temporal retrievers (e.g., TempRetrieval, MRAG) simply degenerate to their base backbone (BGE-M3) in metrics. However, Re^3 distinguishes itself in the final generation phase by the Recency Filter, achieving an accuracy of 0.6339 compared to the base retriever's 0.5900. This suggests that even in the absence of explicit query timestamps, our recency filter contributes by filtering out outdated documents that might otherwise distract the LLM.

Robustness in High-Conflict Environments. The advantages of Re^3 are most pronounced on Re^2 Bench. On the NOAA dataset (weather forecasting), where distinguishing the "latest" forecast from "past" ones is critical, semantic-only retrievers (e.g., BGE-M3) and standard temporal baselines fail significantly (Acc < 0.5). In contrast, Re^3 achieves a remarkable 0.8732 accuracy, doubling the performance of TempRetrieval (0.4238). This massive gap validates the necessity of our Conflict-Aware Recency Filter: while other models retrieve a mix of daily forecasts, Re^3 explicitly arbitrates version conflicts, ensuring the generator receives only the most current evidence.

4.3 Ablation Analysis

Group	Y.Acc	M.Acc	D.Acc	EM
NYC-Q	0.992	0.990	0.991	**0.989**
NYC-D	0.986	0.985	0.983	**0.981**
NOAA-Q	0.985	0.982	0.988	**0.982**
NOAA-D	0.981	0.988	0.985	**0.980**

Table 5: Performance of temporal parsing.

Effect of Temporal Parsing. We compare the timestamps extracted by the Temporal Parsing module against the timestamp information used when generating the questions and documents. We evaluate the parsing quality on NYC and NOAA (representing heterogeneous signals). As shown in Table 5, Acc represents accuracy for time prediction, EM denotes exact-match accuracy over all components. Results are shown for both query-side (Q) and document-side (D) inputs. The parser achieves over 98% Exact Match (EM) accuracy across both datasets. This near-perfect performance validates that our lightweight parsing strategy effectively ensures that downstream modules receive accurate inputs for masking and encoding.

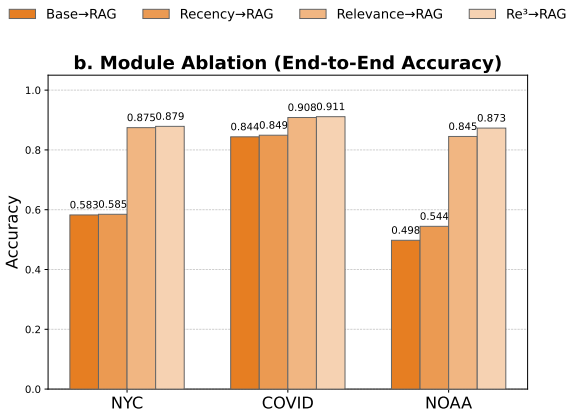

Figure 2: Impact of modules on end-to-end accuracy.

Effect of Relevance & Recency Modules. We further analyze the individual contributions of the two modules. As illustrated in Figure 2, the Relevance Encoder alone provides a substantial accuracy improvement over the base RAG. The Recency Filter is particularly effective in scenarios where outdated or conflicting documents are present (e.g., NOAA). In cases where no outdated evidence exists, it remains largely neutral and does not degrade performance. This highlights their complementary roles: Relevance acts as a coarse-grained semantic-temporal filter, while Recency acts as a fine-grained conflict resolver, suppressing outdated versions that semantically survived the first stage.

4.4 Efficiency and Cost Analysis

A valid concern regarding our Conflict-Aware Recency Filter is the computational overhead of introducing an intermediate LLM. We analyze the trade-off between latency, cost, and performance on the NOAA dataset, comparing standard RAG (retrieval + GPT-4o generation) against Re^3 (retrieval + Llama-3-8B filtering + GPT-4o generation).

As shown in Table 6, although Re^3 incurs additional latency for dual-encoder retrieval (+0.03s) and listwise filtering (+1.66s), this investment yields a 58.2% reduction in input tokens. By purging outdated and conflicting contexts before generation, Re^3 significantly lowers the operational cost (API usage) and accelerates the final generation step (-0.90s). Crucially, the filtering is performed by a locally deployed, lightweight Llama-3-8B, which is sufficient for listwise arbitration without requiring expensive proprietary models. Furthermore, as indicated in our ablation study (Sec. 4.3), the Recency module is detachable; for static knowledge bases with minimal temporal conflicts, it can be bypassed to recover zero-latency overhead, of-

fering a flexible deployment strategy for diverse real-world constraints.

Metric	Std. RAG	Re^3	Δ
Retriever Latency (s)	0.15	0.18	+0.03
Filter Latency (s)	–	1.66	+1.66
Generator Latency (s)	3.04	2.14	**-0.90**
Total Latency (s)	**3.19**	**3.98**	+0.79
Avg. Input Tokens	**991**	**414**	**-58.2%**

Table 6: Cost-benefit analysis on NOAA.

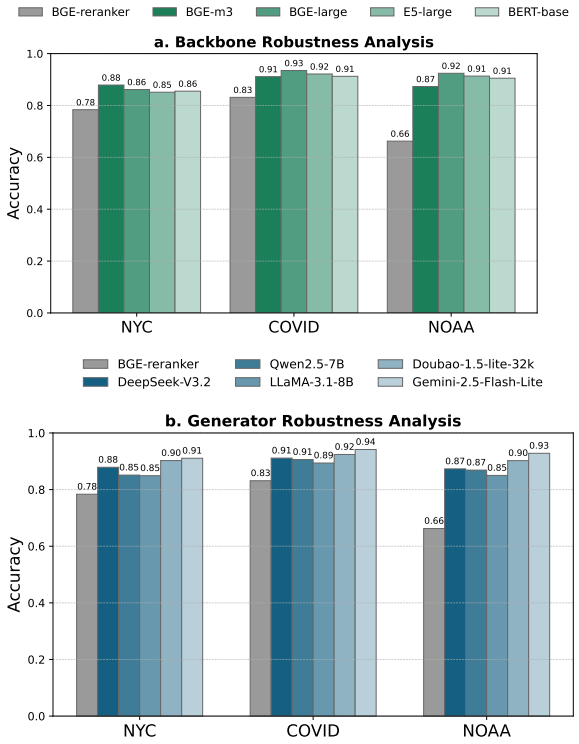

Figure 3: Robustness analysis on Re^2 Bench.

4.5 Robustness Analysis

We further examine the robustness of Re^3 by varying its key components while keeping all other settings fixed. Specifically, we analyze both the retrieval backbone and the generator, and additionally introduce a strong baseline for reference. As shown in Figure 3, we choose BGE-reranker, one of the best-performing base models, as the comparison baseline, and compare it with Re^3 instantiated with different retrieval backbones (e.g., BGE, E5, and BERT) and different generators (e.g., Qwen, LLaMA, Doubao, and Gemini).

The results reveal two clear findings. (i) Re^3 consistently outperforms the BGE-reranker baseline on all three datasets under all tested model variants, demonstrating that the gains of our framework are not tied to a particular model choice. And (ii) although stronger backbones or generators may bring marginal additional improvements, the overall performance trends remain highly consistent. Different backbones lead to only moderate fluctuations, and different generators introduce limited variation without changing the relative ranking pattern across datasets.

Overall, these observations demonstrate that Re^3 is model-agnostic: its effectiveness does not depend on a specific retrieval backbone or generator, but generalizes reliably across different design choices.

5 Conclusion

This work identifies retrieval-side misalignment as a primary cause of temporal hallucinations in RAG. To address this, we propose Re^3, a framework that jointly models relevance and recency to filter incorrect evidence. Re^3 consistently outperforms strong baselines on both public datasets and our new Re^2 Bench, proving effective across diverse backbones and generators. Our findings underscore that principled temporal modeling is essential for building trustworthy RAG systems in dynamic environments.

6 Limitations

The Recency module in our framework operates under the assumption that within credible corpora, the chronologically latest version of conflicting facts supersedes prior ones. While this assumption is reasonable for the official and authoritative datasets used in this work, its applicability in real-world scenarios faces challenges. Defining and identifying "credible" databases is non-trivial, and even official sources may suffer from issues such as data sparsity, delayed updates, or missing records, which could undermine the reliability of our recency-based filtering.

Furthermore, our framework is explicitly optimized for time-sensitive queries targeting the current factual state (i.e., "now-focused" QA). Consequently, a notable limitation is its direct applicability to tasks requiring historical reasoning (e.g., "What was the forecast last week?") or multi-time aggregation, where blindly filtering for recency is inappropriate. Although our underlying Temporal Encoder remains active in such cases to correctly ground queries and documents in the past, handling these scenarios requires relaxing or disabling the

inference-time weighting (α) of the Recency Filter. Future work could explore query-aware routing mechanisms to dynamically adjust this recency bias based on the temporal intent of the query.

Another limitation lies in the computational overhead introduced by the Conflict-Aware Recency Filter. Although we employ a lightweight LLM to process the query and candidate documents in a listwise manner, this step inevitably increases inference latency compared to pure vector retrieval. While we argue that this efficiency trade-off is acceptable for the significant gains in generation faithfulness, it may still pose a bottleneck for high-concurrency or strictly low-latency applications.

Finally, the complexity of temporal parsing remains a challenge. Real-world texts often contain ambiguous temporal signals, such as multiple timestamps within a single sentence or implicit time cues that are difficult to resolve to absolute dates. Errors in temporal extraction can propagate downstream, leading to misalignment in the Time-Aware Encoder or incorrect version arbitration. Additionally, our reliance on publication timestamps (d_{pub}) assumes their availability and reliability. While our contrastive training enables the model to gracefully degrade by leveraging content-based temporal signals when explicit metadata is missing, heavily noisy or manipulated publication dates in open-domain web settings can still compromise the robustness of our filtering mechanism.

7 Ethical Considerations

Our work proposes the Re^3 framework to mitigate temporal hallucinations in RAG systems and introduces the Re^2 Bench, utilizing data from public domains such as NYC motor vehicle collisions, COVID-19 statistics, and NOAA weather forecasts. We have ensured that our use of these datasets complies with their respective licenses and terms of use. We honor the ethical code set out in the ACL Code of Ethics and have prioritized the responsible use of data throughout this research.

Despite the goal of improving factual reliability, there are potential broader impacts to consider. The "latest is correct" heuristic, if applied blindly, carries the risk of amplifying misinformation in scenarios involving data poisoning or timestamp manipulation. Furthermore, the suppression of outdated versions by the Recency Filter might inadvertently obscure historical context or alternative perspectives in tasks that require a comprehensive view of evolving events. Finally, the reliance on LLMs for fact extraction and arbitration inherits the inherent biases and potential for hallucination present in these models, which could lead to systematic errors in evidence selection. We encourage practitioners to implement safeguards, such as cross-referencing multiple sources and maintaining audit trails for filtered content, when deploying such systems in sensitive domains.

As for use of AI Assistants, we used large language models (e.g., ChatGPT) for language polishing and paraphrasing to improve the clarity of our manuscript. All core ideas, methods, experiments, and analyses were conducted by the authors.

Acknowledgments

This work was supported by grants from the Key Technologies R & D Program of Anhui Province (No. 202423k09020039), the National Natural Science Foundation of China (No. 62502486), the Anhui Social Science Research Project (No. AHSKY2022D133), the Fundamental Research Funds for the Central Universities (No. JZ2025HGTB0240 and WK2150110032), and the Guangdong Science and Technology Programme (No. 2025B0101120004).

References

Abdelrahman Abdallah, Bhawna Piryani, Jonas Wallat, Abhishek Anand, and Adam Jatowt. 2025. Tempretriever: Fusion-based temporal dense passage retrieval for time-sensitive questions. *arXiv preprint arXiv:2502.21024*.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, and 75 others. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. In *NeurIPS 2021 Track on Datasets and Benchmarks (Round 2)*. Published: 11 Oct 2021.

Mingyue Cheng, Yucong Luo, Jie Ouyang, Qi Liu, Huijie Liu, Li Li, Shuo Yu, Bohou Zhang, Jiawei

Cao, Jie Ma, and 1 others. 2025. A survey on knowledge-oriented retrieval-augmented generation. *arXiv preprint arXiv:2503.10677*.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 79 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Anoushka Gade, Jorjeta G. Jetcheva, and Hardi Trivedi. 2025. It's about time: Incorporating temporality in retrieval augmented language models. In *Proceedings of the 2025 IEEE Conference on Artificial Intelligence (CAI)*, Santa Clara, CA, USA. IEEE.

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

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2019. Diachronic embedding for temporal knowledge graph completion. *Preprint*, arXiv:1907.03143.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3929–3938.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2).

Jungo Kasai, Keisuke Sakaguchi, yoichi takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, and Kentaro Inui. 2023. Realtime qa: What's the answer right now? In *Advances in Neural Information Processing Systems*, volume 36, pages 49025–49043. Curran Associates, Inc.

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

Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making large language models a better foundation for dense retrieval. *Preprint*, arXiv:2312.15503.

Xiaoyan Li and W. Bruce Croft. 2003. Time-based language models. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, page 469–475, New York, NY, USA. Association for Computing Machinery.

Junda Lin, Zhaomeng Zhou, Zhi Zheng, Shuochen Liu, Tong Xu, Yong Chen, and Enhong Chen. 2026. Vigil: Defending llm agents against tool stream injection via verify-before-commit. *Preprint*, arXiv:2601.05755.

Zirui Liu, Jiatong Li, Yan Zhuang, Qi Liu, Shuanghong Shen, Jie Ouyang, Mingyue Cheng, and Shijin Wang. 2025. am-elo: A stable framework for arena-based llm evaluation. In *International Conference on Machine Learning*, pages 38857–38868. PMLR.

Zirui Liu, Xianquan Wang, Yan Zhuang, Jiatong Li, Qi Liu, Shuanghong Shen, Mingyue Cheng, and Shijin Wang. 2026. Fewer battles, more gain: An information-efficient framework for arena-based LLM evaluation. In *The Fourteenth International Conference on Learning Representations*.

Zirui Liu, Zhuang Yan, Qi Liu, Jiatong Li, Yuren Zhang, Zhenya Huang, Jinze Wu, and Shijin Wang. 2024. Computerized adaptive testing via collaborative ranking. In *Neural Information Processing Systems*.

Edouard Mathieu, Hannah Ritchie, Esteban Ortiz-Ospina, Max Roser, Joe Hasell, Cameron Appel, Charlie Giattino, and Lucas Rodés-Guirao. 2021. A global database of covid-19 vaccinations. *Nature Human Behaviour*, 5:947–953.

National Oceanic and Atmospheric Administration. 2025. National weather service forecast data. `https://www.weather.gov/documentation/services-web-api`. Accessed: 2025-12-09.

New York City Open Data. 2025. Motor vehicle collisions – crashes. `https://data.cityofnewyork.us/Public-Safety/Motor-Vehicle-Collisions-Crashes/h9gi-nx95`. Accessed: 2025-12-01.

Agada Joseph Oche, Ademola Glory Folashade, Tirthankar Ghosal, and Arpan Biswas. 2025. A systematic review of key retrieval-augmented generation (rag) systems: Progress, gaps, and future directions. *arXiv preprint arXiv:2507.18910*.

Jie Ouyang, Tingyue Pan, Mingyue Cheng, Ruiran Yan, Yucong Luo, Jiaying Lin, and Qi Liu. 2025. HoH: A dynamic benchmark for evaluating the impact of outdated information on retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6036–6063, Vienna, Austria. Association for Computational Linguistics.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Benjamin Reichman and Larry Heck. 2024. Dense passage retrieval: Is it retrieving? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13540–13553, Miami, Florida, USA. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Guy D. Rosin, Eytan Adar, and Kira Radinsky. 2017. Learning word relatedness over time. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1168–1178, Copenhagen, Denmark. Association for Computational Linguistics.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding. *Preprint*, arXiv:1807.03748.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. 2024. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, Bangkok, Thailand. Association for Computational Linguistics.

Feifan Wu, Lingyuan Liu, Wentao He, Ziqi Liu, Zhiqiang Zhang, Haofen Wang, and Meng Wang. 2024. Time-sensitive retrieval-augmented generation for question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, pages 2544–2553, New York, NY, USA. Association for Computing Machinery.

Hao Xin, Lei Chen, and Yanyan Shen. 2024. Cost-aware outdated facts correction in the knowledge bases. In *Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA 2024)*, volume 14853 of *Lecture Notes in Computer Science*, pages 257–272. Springer.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*.

Shuo Yu, Mingyue Cheng, Qi Liu, Daoyu Wang, Jiqian Yang, Jie Ouyang, Yucong Luo, Chenyi Lei, and Enhong Chen. 2025. Multi-source knowledge pruning for retrieval-augmented generation: A benchmark and empirical study. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 3931–3941.

Jintao Zhang, Zirui Liu, Mingyue Cheng, Shilong Zhang, Tingyue Pan, Yitong zhou, Qi Liu, and Yanhu Xie. 2025a. Multimodal forecasting of sparse intraoperative hypotension events powered by language model. *Preprint*, arXiv:2505.22116.

Siyue Zhang, Yuxiang Xue, Yiming Zhang, Xiaobao Wu, Anh Tuan Luu, and Chen Zhao. 2025b. MRAG: A modular retrieval framework for time-sensitive question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 3080–3118, Suzhou, China. Association for Computational Linguistics.

Shengming Zhao, Yuchen Shao, Yuheng Huang, Jiayang Song, Zhijie Wang, Chengcheng Wan, and Lei Ma. 2024. Understanding the design decisions of retrieval-augmented generation systems. *arXiv preprint arXiv:2411.19463*.

Zhaomeng Zhou, Lan Zhang, Junyang Wang, and Mu Yuan. 2025. Iot-brain: Grounding llms for semantic-spatial sensor scheduling. *Proceedings of the 2025 ACM Workshop on Access Networks with Artificial Intelligence*.

A Construction Pipeline of Re^2 Bench

We construct the Re^2 Bench datasets through a rigorous pipeline designed to simulate real-world temporal inconsistencies and factual conflicts. The process begins with source-specific ingestion, where raw data from diverse domains (traffic reports for NYC, epidemiological statistics for COVID, and meteorological forecasts for NOAA) is parsed into a unified structured format. This structured data serves as the "ground truth" backbone for our generation process.

A core design principle of our pipeline is the decoupling of temporal context from explicit timestamps. While the structured records contain precise dates (e.g., YYYY-MM-DD), we employ a "temporal masking" strategy during the document generation phase. For a subset of documents, explicit dates are replaced with natural language variations (e.g., "early May", "last Friday", or relative terms), forcing retrieval models to align implicit temporal cues with the query's constraints rather than performing simple string matching.

To rigorously test model robustness against hallucinations and outdated information, we implement a Conflict Injection mechanism. For every positive document-query pair, we programmatically generate "hard negatives"—documents that are semantically highly similar to the positive ground truth but factually contradictory. These contradictions are not random; they are generated by systematically perturbing specific attributes (such as the event date, location, or metric type) while keeping the surrounding context identical. This ensures that the negatives are "plausible but false," mimicking the challenging retrieval scenarios found in large-scale web indices.

Finally, we utilize Large Language Models (LLMs) to bridge the gap between structured data and natural language. We employ a dual-prompting strategy: one set of prompts transforms structured records into narrative-style "documents" (simulating news reports, logs, or bulletins), while a separate, distinct set of prompts generates "user queries" that seek specific information contained within those documents. This separation ensures linguistic diversity and prevents the model from learning trivial artifacts between the document and query distributions. The overall procedure is formalized in Algorithm 1.

Algorithm 1 Data Generation and Conflict Injection Pipeline

Require: Raw records $\mathcal{R}$, LLM generator $\mathcal{G}$
$\mathcal{D} \leftarrow \emptyset$, $\mathcal{Q} \leftarrow \emptyset$
for each event cluster C in $\mathcal{R}$ **do**
  $r^+ \leftarrow \textsc{SelectPositive}(C)$
  Step 1: Data Normalization & Masking
  $T_{\text{mask}} \leftarrow \textsc{GenerateDateVariants}(\textsc{Date}(r^+))$
  $d^+ \leftarrow \mathcal{G}(\textsc{Paraphrase}, r^+, T_{\text{mask}})$
  Step 2: Conflict Injection
  $N \leftarrow \emptyset$
  $r_t^- \leftarrow \textsc{Perturb}(r^+, \text{date shift } \Delta t)$
  $r_e^- \leftarrow \textsc{Perturb}(r^+, \text{attr} \in \{\text{location}, \text{metric}\})$
  for r^- in $\{r_t^-, r_e^-\}$ **do**
    $d^- \leftarrow \mathcal{G}(\textsc{Paraphrase}, r^-)$
    $N \leftarrow N \cup \{d^-\}$
  end for
  Step 3: LLM-driven QA Synthesis
  $q \leftarrow \mathcal{G}(\textsc{QueryGen}, \textsc{TargetFact}(r^+))$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{d^+\} \cup N$
  $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(q, d^+, N)\}$
end for

A.1 Dataset: Re^2 Bench-NYC

Data Source. We source traffic incident data from the "Motor Vehicle Collisions - Crashes" database, managed by the New York Police Department (NYPD)[2] and hosted on NYC Open Data. This comprehensive dataset records details of vehicle collisions in New York City, including the date, time, borough, street names, and specific casualty counts (pedestrians, cyclists, and motorists). The raw data is updated daily and contains over 2 million records with 29 attributes per entry.

Preprocessing Pipeline. To ensure data quality and consistency for retrieval tasks, we implement a specialized cleaning pipeline that transforms raw CSV records into structured JSON objects.

1. **Schema Normalization:** The raw CSV schema is non-stationary, with column names varying across versions (e.g., "CRASH DATE" vs. "crash_date"). We employ a mapping strategy to unify these fields into a canonical schema.

2. **Field Parsing & Validation:**

   - *Temporal Standardization:* Original date strings (e.g., "MM/DD/YYYY") are

[2] https://opendata.cityofnewyork.us/

	Content
Query	How many pedestrians were injured in the collision at **Brooklyn** on **July 4, 2024**?
Positive Doc	On the layout of Independence Day, **July 4, 2024**, a traffic incident occurred in **Brooklyn**. Police reports confirm that **2 pedestrians** sustained injuries during the crash.
Hard Negative	Regarding the accident in **Brooklyn** on **July 14, 2024**, authorities noted that no pedestrians were harmed, although vehicle damage was reported.

Table 7: **Example from Re^2 Bench-NYC.** The negative document shares the same location and topic but refers to a different date (Time Hallucination).

parsed and standardized to ISO 8601 format ("YYYY-MM-DD"). Records with missing or malformed dates are discarded.

- *Casualty Aggregation:* We extract eight distinct injury/fatality metrics (e.g., `number_of_pedestrians_injured`, `number_of_cyclist_killed`) and cast them to integers, treating missing values as zeros.
- *Location Formatting:* We consolidate address fields (`borough`, `on_street`, `cross_street`) to construct a complete location description.

3. **Filtering:** Entries lacking a unique `collision_id` or valid timestamp are removed to maintain the integrity of the ground truth.

The resulting cleaned dataset serves as the structured knowledge base for the subsequent text generation phase. An example is shown in Table 7.

A.2 Dataset: Re^2 Bench-COVID

Data Source & Archival Records. Re^2 Bench-COVID is constructed from country-level COVID-19 time-series data curated by Our World in Data (OWID), which aggregates official public health reports into a unified daily record format.[3] Unlike the continuously evolving forecasts in Re^2 Bench-NOAA, the COVID dataset represents archival records, where each data point corresponds to a finalized observation for a specific country on a specific calendar date. Each daily record is indexed by a `(country, date)` pair and contains a collection of numeric indicators, including confirmed cases, deaths, testing statistics, vaccination coverage, and related epidemiological attributes.

Data Normalization & Metric Taxonomy. We first normalize the raw OWID records into a unified and structured representation to support reliable downstream generation. All dates are standardized to the ISO format (`YYYY-MM-DD`), and numeric indicators are retained as floating-point values. To facilitate controlled semantic perturbations, we further organize all indicators into semantically coherent metric groups (e.g., cases, deaths, testing, and vaccinations), yielding a consistent metric taxonomy across countries and time. The resulting cleaned dataset serves as the canonical source for subsequent sample construction.

Fact Anchoring & Index Organization. To enable systematic construction of hard negatives, we treat each atomic epidemiological statement as a grounded fact represented by a tuple $\langle c, t, m, v \rangle$, corresponding to a country c, a date t, a metric m, and its reported value v. We organize the cleaned records such that, for any anchored fact, we can efficiently retrieve alternative valid countries, dates, or metrics that share partial structure with the ground truth. This organization allows us to inject controlled single-attribute conflicts while preserving high topical overlap between positive and negative documents.

Positive and Hard Negative Construction. For each anchored fact, we generate one positive document and three hard negative documents via *single-attribute conflict injection*:

- **Country-Conflicting Negative:** The country is replaced with another country that reports the same metric on the same date, while the date and metric remain unchanged.
- **Date-Conflicting Negative:** The date is replaced with another valid reporting date for the same country and metric, while preserving the country and metric.
- **Metric-Conflicting Negative:** The metric is replaced with another semantically related indicator from the same metric group (e.g., both from `cases`), while keeping the country and date fixed.

[3] OWID COVID-19 Data Documentation

	Content
Query	In **Gibraltar** on **August 25, 2022**, what was the seven-day rolling average of newly confirmed COVID-19 cases per day?
Positive Doc	In late August 2022, **Gibraltar**'s COVID-19 situation showed signs of stabilization. On **August 25, 2022**, the seven-day rolling average of new confirmed cases per day stood at 2.143, reflecting a low but persistent level of community transmission. This figure indicated that while the territory had largely controlled major outbreaks, continued vigilance was necessary.
Hard Negative 1	On **August 25, 2022**, **Somalia** reported no new COVID-19 cases in its seven-day rolling average. The country's official data showed a value of 0.0 for the average daily new confirmed cases, likely reflecting limited testing capacity and reporting challenges in the region. This low figure contrasted with higher transmission rates observed in other parts of the world at that time.
Hard Negative 2	By **July 25, 2023**, **Gibraltar**'s pandemic situation had improved significantly. The seven-day average of new confirmed cases per day had dropped to 0.3, indicating that active transmission was no longer being detected in the community. This milestone reflected successful vaccination campaigns and natural immunity buildup over the preceding year.

Table 8: **Example from Re^2 Bench-COVID.** The negative documents present the same location or time.

All negative documents remain topically plausible and structurally similar to the positive document, yet are factually inconsistent with respect to the queried fact.

LLM-Driven Document & Query Synthesis. We employ a Large Language Model to transform structured fact tuples into natural-language documents and queries. For each fact, the LLM generates one positive paragraph and three aligned hard negatives as described above. Queries are synthesized to explicitly mention the target country and to express temporal constraints using diverse surface forms (e.g., "July 21, 2020", "2020-07-21", or "07/21/2020"), encouraging robustness to temporal expression variability.

Output Format & Dataset Splitting. Each sample consists of a query with its identifier, timestamp, and ground-truth answer, together with references to one positive document and three hard negatives, all annotated with structured metadata (country, date, metric, and value). The dataset is finally partitioned into training and test splits using a fixed ratio, and each split includes exactly the documents referenced by its corresponding queries.

An example is shown in Table 8

A.3 Dataset: Re^2 Bench-NOAA

Data Source & Real-Time Ingestion. Unlike the archival nature of the NYC and COVID datasets, Re^2 Bench-NOAA is constructed from live meteorological data to capture the dynamic evolution of weather forecasts. We utilize the National Oceanic and Atmospheric Administration (NOAA) National Weather Service API[4] to fetch real-time grid-based forecasts. The data collection process is automated via a Python script (`generate_dataset.py`) that iterates through a predefined list of major U.S. cities (mapped to their respective grid coordinates `gridId`, `gridX`, `gridY`). For each city per execution cycle, we retrieve a 7-day forecast window, extracting daily summaries that include the `shortForecast` (e.g., "Partly Sunny") and `temperature` (in Fahrenheit). This process allows us to capture multiple versions of a forecast for the same target date as they evolve over time (e.g., a forecast for December 20th generated on December 14th vs. one generated on December 18th), providing a rich basis for testing temporal versioning conflicts.

Cluster Construction & Versioning. To organize this streaming data, we group records into unique clusters keyed by `(city, target_date)`. Within each cluster, multiple forecast versions are sorted by their `publication_date` (the timestamp when the API was queried). This structure is critical for conflict injection: the most recent forecast version typically serves as the "ground truth" (Positive), while earlier, outdated versions for the same target date serve as high-quality "outdated" negatives (Hard Negatives).

LLM-Driven Document & Query Synthesis. We employ a two-stage generation process using Large Language Models to transform structured API responses into natural language text:

1. **Document Generation:** For each structured forecast record (containing `city`, `forecast_date`, `weather`, `temperature`), we prompt the LLM to write a short weather report. To prevent trivial pattern matching, we enforce diversity in temporal expressions (e.g., mapping "2025-12-12" to "Friday, De-

[4] https://api.weather.gov

	Content
Query	On **November 5, 2025**, what will the weather be like in **Pontiac**?
Positive Doc	**Issued just one day prior, on November 4, 2025**, NOAA's updated forecast for **Pontiac** on **11/05/2025** indicates a slight warming trend. The prediction calls for a continued chance of light rain, but with temperatures reaching around 60°F. This more recent forecast offers a refined look at the day's expected conditions.
Outdated Doc	NOAA's weather forecast for **Pontiac**, **issued on October 31, 2025**, predicts conditions for **November 5, 2025**. The outlook includes a chance of light rain with an expected high temperature of 58°F. This early forecast provides a preliminary glimpse at the potential weather for that date.
Hard Negative 1	**Issued on August 3**, this NOAA forecast indicates that **Pontiac** will see mostly sunny skies on **08/08/2025** before a slight chance of showers and thunderstorms emerges. Temperatures are forecast to climb to a high of 88°F.
Hard Negative 2	A NOAA forecast **released on November 4** calls for mostly sunny weather in **Port Huron** on **November 5**. The anticipated high for the day is 89°F.

Table 9: **Example from Re^2 Bench-NOAA.** The negative documents present confused location, time or outdated publication time.

cember 12th" or "next Friday") and sentence structures.

2. **Query Generation (Dual-Type):** We generate two distinct types of queries to test different retrieval capabilities:
    - **Type A (Explicit Date):** Queries that specify a precise target date (e.g., "What is the weather forecast for Boston on December 12th?"). These test the model's ability to align specific temporal constraints.
    - **Type B (Implicit/Latest):** Queries that ask for the most current information without a specific date (e.g., "What is the latest weather outlook for Miami?"). These require the model to implicitly prioritize the most recent document version based on metadata or context.

Conflict Injection Strategy. We rigorously construct negative samples to challenge the retriever:

- **Time Hallucination:** Documents describing the weather for the same city but on a different date (e.g., querying for Monday but retrieving Tuesday's forecast).
- **Event Hallucination:** Documents describing the same date but for a different city (e.g., retrieving Chicago's weather when asking about New York).
- **Outdated Information:** For Type B queries, we specifically include earlier forecast versions for the same city/date pair as negatives, testing the model's ability to discern the most up-to-date information.

The final dataset consists of JSON-formatted tuples containing the query, the positive document, and a set of hard negatives, all timestamped with both their validity date (forecast target) and publication date. And publication time will be injected into documents after retrieval, to ensure information of recency is accessed to generator. An example is shown in Table 9, as documents are presented as full message (with publication time).

A.4 Prompt Templates for Data Generation

To ensure reproducibility and high-quality alignment between positive and negative samples, we employ a batched generation strategy for documents. Instead of generating documents individually, we provide the LLM with a cluster of related records (one positive and multiple perturbed negatives) in a single context. As illustrated in Figure 11 and Figure 10, the structured prompt explicitly encodes the entity, temporal constraint, and information requirement for each record. Building on this query formulation, the document-generation prompt groups the positive instance with its corresponding perturbed negatives, forcing the model to maintain stylistic consistency across the batch while accurately reflecting subtle factual variations (e.g., date or value changes) specified in each record.

B Computational Cost Analysis

We analyze the computational overhead introduced by the Re^3 framework. A primary concern with multi-stage RAG pipelines is the potential latency penalty. However, our experimental results on the NOAA dataset demonstrate that Re^3 offers a highly favorable trade-off: a marginal increase in end-to-

end latency in exchange for substantial reductions in token consumption and operational costs.

B.1 Experimental Setup

We evaluated the performance metrics using average values derived from the NOAA dataset experiments. The baseline (Standard RAG) feeds all top-k retrieved documents directly into the generator (GPT-4o). In contrast, Re^3 employs a parallelized Llama-3-8B model for the intermediate filtering step, passing only the verified, non-conflicting candidates to the generator.

B.2 Case Study for Public Datasets

We examine the qualitative performance of Re^3 on public datasets to illustrate its capability in handling both static temporal alignment and dynamic information evolution.

Precise Temporal Alignment (TimeQA & Nobel-TSRAG). For datasets requiring strict temporal matching, such as TimeQA and Nobel-TSRAG, the challenge lies in distinguishing the target event from temporally adjacent but irrelevant facts. As shown in Figure 5, when the user queries the position of Carl Eric Almgren in *1960*, Re^3 successfully prioritizes documents containing the exact timestamp "1960". Notably, the model is not misled by Document 2, which describes a different role held by the same entity in *1962*, demonstrating robust resistance to semantic similarity when temporal constraints are unmet. Similarly, in the Nobel-TSRAG case (Figure 6), the system accurately locks onto the 1901 laureate, distinguishing it from winners in 1911 and 1921. This confirms that Re^3 effectively integrates timestamps as a primary feature for relevance ranking.

Filtering Outdated Information (HoH). The HoH dataset serves as a testbed for handling evolving facts. Figure 7 provides a critical insight into the behavior of our Recency-aware filtering mechanism. Initially, the retrieval set contains conflicting information regarding the movie's Rotten Tomatoes rating: Document 1 (published 2024-09-01) states a 17% rating, while Document 2 (published 2024-08-01) reports an outdated 20% rating. After applying the recency filtering of Re^3, the outdated Document 2 is successfully removed, resolving the factual conflict before it reaches the LLM.

Crucially, this filtering process is selective. As observed in the transition from "Before Recency" to "After Recency" results, non-conflicting documents (such as Document 3 regarding a different film/review) are preserved. This demonstrates that Re^3 does not indiscriminately penalize older documents; rather, it specifically targets *outdated versions of the same information*, ensuring the LLM receives the most current evidence without losing broader context.

B.3 Case Study for Re^2 Bench-NOAA

The NOAA dataset presents a uniquely challenging scenario characterized by high-frequency updates and predictive uncertainty. Unlike historical facts that eventually settle on a ground truth, weather forecasts are continuously refined as the target date approaches. We analyze two representative cases to demonstrate the capabilities and current limitations of Re^3 in this dynamic environment.

Robustness Against Spatial and Temporal Noise (Case B). Figure 9 illustrates a successful retrieval and reasoning process for the query regarding Pontiac's weather. The initial retrieval phase introduces two types of noise: (1) *Spatial Distractors*: Documents related to "Pocatello", "Chattanooga", and "Pawtucket" are retrieved due to phonetic or semantic overlap; (2) *Temporal Conflicts*: Multiple forecasts for "Pontiac" exist, issued on October 31 (58°F) and November 4 (60°F).

As shown in the "After Recency" results, the Re^3 module effectively addresses the temporal conflict by filtering out the outdated October 31 forecast, retaining only the most recent November 4 update for Pontiac. Notably, Re^3 preserves the spatial distractors (e.g., Pocatello) because they do not constitute a *temporal* conflict with the target entity. The final success relies on the LLM's ability to distinguish the correct entity from the remaining spatial distractors, provided that Re^3 has successfully purified the temporal dimension. This case demonstrates the synergy between the retriever's temporal filtering and the generator's semantic understanding.

Analysis of Fine-Grained Update Failure (Case A). Figure 8 presents a failure case that highlights the difficulty of handling fine-grained numerical updates. The ground truth (Partly Sunny, 48°F) corresponds to the final forecast issued on November 28 (Doc 1). However, the model outputs "Partly sunny with a high near 49°F," matching the slightly older forecast from November 27 (Doc 2).

The root cause of this error lies in the granularity of the recency filtering. Because the two documents are published only one day apart and share high semantic similarity, the Recency module retains both as valid evidence. Unlike clear-cut contradic-

tions (e.g., "Winner A" vs. "Winner B"), the subtle numerical shift (48°F vs. 49°F) creates ambiguity for the LLM. Faced with two highly credible, recent sources, the model failed to strictly adhere to the "most recent" instruction, instead blending information or selecting the document with potentially more salient phrasing. This suggests that for high-frequency data streams like NOAA, a stricter "Top-1" temporal filtering strategy or more aggressive penalty for near-duplicate timestamps may be necessary.

C Effect of Temporal Encoding Strategy

We present the efficiency of temporal representation by comparing it with two methods: (1) Direct: Concatenating normalized raw values to semantic embedding; (2) Absolute: Applying positional encoding to timestamps before concatenation.

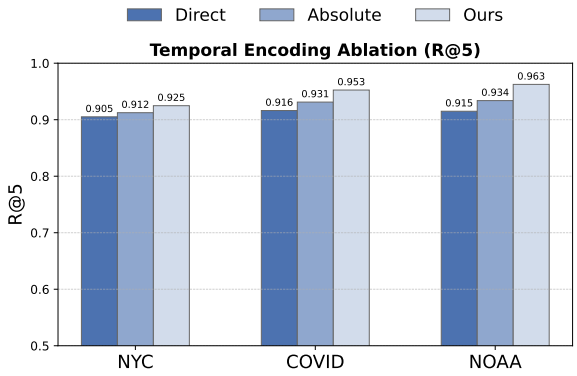

Figure 4: Comparison of temporal encoding strategies.

As shown in Figure 4, our proposed encoding consistently achieves the highest Recall@5 across all datasets. While Absolute encoding moderately improves over Direct by learning specific time-bucket representations, it struggles to generalize to unseen timestamps. In contrast, our method combines normalized linear projections with sinusoidal embeddings, enabling the model to capture both long-term evolution and fine-grained cyclic patterns. This confirms that a hybrid, continuous temporal representation is essential for high-precision time-sensitive retrieval.

Varying λ_{hard} (fix $\lambda_{\text{tw}} = 0.1$)	**Recall@5**	**MRR**
$\lambda_{\text{hard}} = 0.0$	0.905	0.872
$\lambda_{\text{hard}} = 0.1$	**0.924**	**0.891**
$\lambda_{\text{hard}} = 0.2$	0.920	0.886
$\lambda_{\text{hard}} = 0.5$	0.895	0.862
Varying λ_{tw} (fix $\lambda_{\text{hard}} = 0.1$)	**Recall@5**	**MRR**
$\lambda_{\text{tw}} = 0.0$	0.901	0.864
$\lambda_{\text{tw}} = 0.1$	**0.924**	**0.891**
$\lambda_{\text{tw}} = 0.2$	0.919	0.881
$\lambda_{\text{tw}} = 0.5$	0.893	0.858

Table 10: Sensitivity analysis of the hyperparameters λ_{hard} and λ_{tw} on the Re^2 Bench–NYC dataset.

D Sensitivity Analysis of Hyperparameters λ

Our training objective for the Time-Aware Dual Relevance Encoder introduces exactly two hyperparameters: λ_{hard}, which weights the margin-based loss for general hard negatives, and λ_{tw}, which weights the margin-based loss specifically for time-wrong negatives.

To investigate the impact of these hyperparameters on retrieval performance, we conduct a comprehensive sensitivity analysis on the Re^2 Bench–NYC dataset, chosen for its moderate size and representative temporal challenges.

In our experiments, we vary one hyperparameter across the set $\{0.0, 0.1, 0.2, 0.5\}$ while fixing the other at the optimal value of 0.1. The resulting Recall@5 and MRR metrics are reported in Table 10.

Analysis and Discussion. Several key observations can be drawn from the results:

- **Ablation Validation ($\lambda = 0$):** Removing either margin-based regularizer ($\lambda_{\text{hard}} = 0$ or $\lambda_{\text{tw}} = 0$) results in a noticeable performance drop of 2–3% in MRR. This confirms that both the general semantic hard negatives and our explicitly constructed time-wrong negatives are essential for learning robust temporal-semantic representations.

- **Robustness in Moderate Ranges:** The framework exhibits strong robustness when the hyperparameters are kept within the moderate range of $[0.1, 0.2]$. The performance fluctuations between 0.1 and 0.2 are minimal, indicating that our model does not require exhaustive hyperparameter tuning to achieve near-optimal results.

- **Over-regularization ($\lambda = 0.5$):** When the weights are increased excessively (e.g., to 0.5), we observe a significant degradation in performance ($\sim$3% drop in MRR). We attribute this to over-regularization: forcing the model to focus too heavily on distinguishing hard/time-wrong negatives disrupts the underlying semantic alignment (the InfoNCE loss), leading to a distorted embedding space.

Overall, setting both λ_{hard} and λ_{tw} to 0.1 provides the optimal trade-off between semantic matching and temporal discrimination.

Case Study on TimeQA

Question:
What position did Carl Eric Almgren hold In the period of 1960?

Answer:
commander of Jämtland Ranger Regiment

Re^3 Retrieval Results:
1. In the period of 1960, Carl Eric Almgren served as the commander of Jämtland Ranger Regiment.
2. Throughout 1962, Carl Eric Almgren was appointed as the military commander of the Eastern Military District.
3. According to records from 1960, George Koltanowski was awarded the title of honorary Grandmaster.
4. Aurelio Lampredi worked for Fiat Notably, in 1960.
5. Nobby Stiles was a member of Manchester United Back in 1960.

Input to LLM:
`You are an intelligent assistant task with answering time-sensitive questions based on provided reference materials...`

LLM Answer:
commander of Jämtland Ranger Regiment

Figure 5: A case study from the TimeQA.

Case Study on Nobel-TSRAG

Question:
Who won the Nobel Prize in Literature around 1901?

Answer:
Sully Prudhomme

Re^3 Retrieval Results:
1. In 1901, Sully Prudhomme was awarded the Nobel Prize in Literature.
2. During 1901, Henry Dunant was recognized as a laureate of the Nobel Prize in Nobel Prize.
3. In 1901, Frédéric Passy was awarded the Nobel Prize in Nobel Prize.
4. During 1911, Maurice Maeterlinck was awarded the Nobel Prize in Literature.
5. Anatole France was honored with the Nobel Prize in Literature in 1921.

Input to LLM:
```
You are an intelligent assistant task with answering time-sensitive questions
based on provided reference materials...
```

LLM Answer:
Sully Prudhomme

Figure 6: A case study from the Nobel-TSRAG.

Case Study on HoH

Question:
What is the approval rating of "Hollow Man 2" on Rotten Tomatoes?

Answer:
17%

Re^3 Retrieval Results before Recency:
1. "Hollow Man 2" received a mixed response from critics, achieving a 17% approval rating on Rotten Tomatoes based on 6 reviews. Publication time: 2024-09-01
2. "Hollow Man 2" received a mixed response from critics, achieving a 20% approval rating on Rotten Tomatoes based on 5 reviews. Publication time: 2024-08-01
3. The film has a 95% approval rating based on 60 reviews on Rotten Tomatoes with an average rating of 7.8/10 and with the site's consensus saying: "Wes Craven's intelligent premise... Publication time: 2024-10-01
4. Two Evil Eyes" holds a rating of 63% on Rotten Tomatoes based on 19 reviews, with an average rating of 5.4/10. Publication time: 2024-08-01
5. The film has a 93% approval rating based on 60 reviews on Rotten Tomatoes with an average rating of 7.8/10 and with the site's consensus saying: "Wes Craven's intelligent premise... Publication time: 2024-11-01

Re^3 Retrieval Results after Recency:
1. "Hollow Man 2" received a mixed response from critics, achieving a 17% approval rating on Rotten Tomatoes based on 6 reviews. Publication time: 2024-09-01
2. Two Evil Eyes" holds a rating of 63% on Rotten Tomatoes based on 19 reviews, with an average rating of 5.4/10. Publication time: 2024-08-01
3. The film has a 93% approval rating based on 60 reviews on Rotten Tomatoes with an average rating of 7.8/10 and with the site's consensus saying: "Wes Craven's intelligent premise... Publication time: 2024-11-01

Input to LLM:
`You are an intelligent assistant task with answering time-sensitive questions based on provided reference materials...`

LLM Answer:
17%

Figure 7: A case study from the HoH.

Case Study on NOAA A

Question:
How will the weather be in Shreveport on November 30, 2025?

Answer:
Partly Sunny, 48°F

Re^3 Retrieval Results before Recency:
1. Shreveport's weather for November 30, according to the NOAA forecast published on November 28, is expected to be partly sunny. The predicted high temperature is 48°F. This final outlook before the target date indicates a cool and partly cloudy day.
2. NOAA's weather forecast for Shreveport on November 30, 2025, issued on November 24, indicates a chance of showers and thunderstorms. The expected high temperature is around 69°F. This early outlook suggests a mild but potentially wet day ahead.
3. The NOAA forecast released on November 27 for Shreveport details the weather for November 30. It starts with a slight chance of rain showers in the morning, becoming partly sunny later, with a high near 49°F. This suggests a clearing trend through the day.
4. Issued on November 25, a NOAA forecast predicts Shreveport will experience a chance of showers and thunderstorms on 11/30/2025. Temperatures are anticipated to reach approximately 59°F. The forecast reflects a slight cooling trend from the previous day's prediction.
5. For the date 2025-11-30, NOAA provides a forecast for Shreveport calling for a chance of rain showers. Published on November 26, it estimates the daytime temperature will be near 54°F. This update shows a continued adjustment toward cooler conditions.

Re^3 Retrieval Results after Recency:
1. Shreveport's weather for November 30, according to the NOAA forecast published on November 28, is expected to be partly sunny. The predicted high temperature is 48°F. This final outlook before the target date indicates a cool and partly cloudy day.
2. The NOAA forecast released on November 27 for Shreveport details the weather for November 30. It starts with a slight chance of rain showers in the morning, becoming partly sunny later, with a high near 49°F. This suggests a clearing trend through the day.

Input to LLM:
```
You are an intelligent assistant task with answering time-sensitive questions
based on provided reference materials...
```

LLM Answer:
Partly sunny with a high near 49°F.

Figure 8: A case study from the NOAA, which is failure

Case Study on NOAA B

Question:
On November 5, 2025, what will the weather be like in Pontiac?

Answer:
Chance Light Rain, 60°F

Re^3 Retrieval Results before Recency:
1. NOAA's weather forecast for Pontiac, issued on October 31, 2025, predicts conditions for November 5, 2025. The outlook includes a chance of light rain with an expected high temperature of 58°F. This early forecast provides a preliminary glimpse at the potential weather for that date.
2. Issued just one day prior, on November 4, 2025, NOAA's updated forecast for Pontiac on 11/05/2025 indicates a slight warming trend. The prediction calls for a continued chance of light rain, but with temperatures reaching around 60°F. This more recent forecast offers a refined look at the day's expected conditions.
3. According to NOAA, the weather forecast for Pocatello on November 5, 2025, predicts a day that will start out mostly cloudy before transitioning to a chance of rain showers. The high temperature is expected to reach 63°F. This forecast was issued on November 3, 2025.
4. According to NOAA, the weather forecast for Chattanooga on November 5, 2025, predicts a sunny day with temperatures reaching 70°F. This forecast was issued on October 31, 2025, providing an early outlook for the conditions expected in early November.
5. Issued on November 3, 2025, the NOAA forecast indicates that Pawtucket will experience mostly cloudy skies on 11/05/2025. Temperatures are expected to reach around 58°F, making it a slightly warmer day compared to earlier predictions.

Re^3 Retrieval Results after Recency:
1. Issued just one day prior, on November 4, 2025, NOAA's updated forecast for Pontiac on 11/05/2025 indicates a slight warming trend. The prediction calls for a continued chance of light rain, but with temperatures reaching around 60°F. This more recent forecast offers a refined look at the day's expected conditions.
2. According to NOAA, the weather forecast for Pocatello on November 5, 2025, predicts a day that will start out mostly cloudy before transitioning to a chance of rain showers. The high temperature is expected to reach 63°F. This forecast was issued on November 3, 2025.
3. According to NOAA, the weather forecast for Chattanooga on November 5, 2025, predicts a sunny day with temperatures reaching 70°F. This forecast was issued on October 31, 2025, providing an early outlook for the conditions expected in early November.
4. Issued on November 3, 2025, the NOAA forecast indicates that Pawtucket will experience mostly cloudy skies on 11/05/2025. Temperatures are expected to reach around 58°F, making it a slightly warmer day compared to earlier predictions.

Input to LLM:
`You are an intelligent assistant task with answering time-sensitive questions based on provided reference materials...`

LLM Answer:
a chance of light rain with temperatures reaching around 60°F

Figure 9: A case study from the NOAA, which is successful.

Listwise Fact Extraction Prompt

You are an expert information extraction assistant. Your task is to extract relevant factual details from a list of retrieved documents to help answer a time-sensitive question.

Instructions:

1. You are given a question and a list of retrieved candidate documents. Each document is accompanied by its publication date and an index.
2. Extract the core factual claim from EACH document that directly addresses the question.
3. Structured Output: For each document, identify the underlying entity (Head), the property being queried (Relation), and the specific value stated in that document (Value).
4. If a document does not contain information relevant to the question, output "NULL" for its value.
5. Entity Resolution: Ensure that variations of the same entity (e.g., "NYC" and "New York City") are normalized to a consistent Head name.

Output Format:

Return strictly a JSON array of objects, where each object corresponds to a candidate document.
Format: [{"doc_index": 0, "head": "...", "relation": "...", "value": "..."}, ...]
- MUST output valid JSON only.
- Do NOT include markdown blocks, reasoning, or conversational filler.

Input Data:

Question: {Question from dataset}
Retrieved Candidates:
[0] Date: {PubDate_0}): {Document_0}
[1] Date: {PubDate_1}): {Document_1}
...

Figure 10: Prompt template used for the listwise fact extraction within the Conflict-Aware Recency Filter. This instruction guides the LLM to uniformly extract structured factual triples (Head, Relation, Value) from multiple retrieved candidate documents, ensuring consistent version comparison and temporal conflict resolution. The full reproduction scripts are available in our open-source repository.

Query Generation Prompt

You are a user searching for specific historical information. Formulate a search query based on the provided target fact.

Input Context Target Fact:
- Entity: {ENTITY}
- Date: {DATE}
- Information Needed: {METRIC_DESCRIPTION} (Value: {VALUE})

Task Requirements
1. Write ONE search query that a user would type to find this specific value.
2. Temporal Constraint: You MUST mention the date or time range explicitly using the phrase "{DATE_VARIANT}".
3. Entity Constraint: You MUST mention the entity {ENTITY}.
4. Style: The query should be concise and interrogative (e.g., starts with "How many", "What was", or "Report on...").
5. Privacy: Do NOT reveal the actual numeric value ({VALUE}) in the query.

Output Format
Return strictly a JSON object: {"query": "YOUR_GENERATED_QUERY"}

Documents Generation Prompt

You are an expert reporter. You will receive a list of structured event records that share a common context but differ in specific details (e.g., dates, locations, or values). Your task is to write a unique, independent narrative paragraph for EACH record.

Input Context: Here is a group of related records:
Record 0 (Positive): {STRUCTURED_DATA_0}
Record 1 (Negative - Time Conflict): {STRUCTURED_DATA_1}
Record 2 (Negative - Entity Conflict): {STRUCTURED_DATA_2}...

Task Requirements
1. Independent Generation: For each record index, write one narrative paragraph (4–6 sentences) that strictly adheres to that record's data.
2. Stylistic Consistency: Use a similar tone and structure for all paragraphs to ensure they look like reports from the same source.
3. Factual Precision: Accurately reflect the {date}, {location}, and {metric_value} for each specific record. Do NOT mix information between records.
4. Masking: For Record 0, refer to the date using the natural language variant "{DATE_VARIANT_0}". For others, use their respective variants.
Output Format
Return a JSON array: [{"index": 0, "text": "..."}, {"index": 1, "text": "..."}, ...]

RAG Answering Prompt

You are an intelligent assistant task with answering time-sensitive questions based on provided reference materials.

Instructions:
1. You are given a question and a list of retrieved documents. Each document includes a publication date.
2. Answer the question based ONLY on the provided documents.
3. Temporal Conflict Resolution: If there is factual conflict among documents (e.g., a statistic has changed over time), you must prioritize the evidence from the most recent, non-obsolete document.
4. If the provided documents do not contain sufficient information to answer the question, output "NOT SURE".

Output Format:
- Your final output MUST be the exact answer string only.
- Do NOT include prefixes (e.g., "The answer is..."), reasoning, or punctuation.
- Do NOT add any introductory or concluding remarks.

Input Data:
Question: {Question from datasets}
Retrieved Documents: {Documents from retrieval}

Figure 11: Prompt templates used for benchmark query generation, document generation, and RAG answering.