When Facts Change: Probing LLMs on Evolving Knowledge with evolveQA

Anonymous ACL submission

Abstract

LLMs often fail to handle temporal knowledge conflicts–contradictions arising when facts evolve over time within their training data. Existing studies evaluate this phenomenon through benchmarks built on structured knowledge bases like Wikidata, but they focus on widely-covered, easily-memorized popular entities and lack the dynamic structure needed to fairly evaluate LLMs with different knowledge cut-off dates. We introduce evolveQA, a benchmark specifically designed to evaluate LLMs on temporally evolving knowledge, constructed from 3 real-world, time-stamped corpora: AWS updates, Azure changes, and WHO disease outbreak reports. Our framework automatically identifies naturally occurring knowledge evolution and generates questions with gold answers tailored to different LLM knowledge cut-off dates. Through extensive evaluation of 12 open and closed-source LLMs across 3 knowledge probing formats, we demonstrate significant performance drops of up to 31% on evolveQA compared to static knowledge questions.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across diverse tasks, from text generation to complex reasoning (Brown et al., 2020; Wei et al., 2021; Chang et al., 2024; Su et al., 2024a). However, their reliability faces a critical challenge: *temporal knowledge conflicts*, where factual information evolves over time, creating contradictions between outdated and current versions within the model's training data. This problem is pervasive across domains such as healthcare, where guidelines update with new research; technology, where specifications change with software releases, and so on. As LLMs increasingly serve as knowledge sources in high-stake applications, their tendency to recall outdated information over recent updates, as illustrated in Figure 1, poses significant risks to system reliability and user trust.

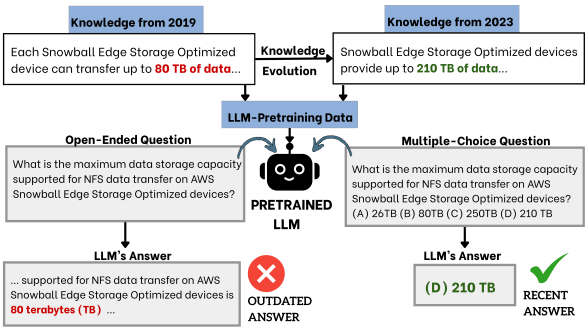

Figure 1: LLMs often give outdated answers to open-ended questions, even though they can identify correct and current information when given multiple-choice options; this suggests that while LLMs hold more recent knowledge in their parameters, they struggle to recall it without explicit prompting.

Prior works studied broadly three types of knowledge conflicts in LLMs: conflicts in external retrieved documents (Pan et al., 2021; Chen et al., 2022; Wang et al., 2025), conflicts between external context and model's internal knowledge (Kortukov et al., 2024; Wang et al., 2023; Xie et al., 2023; Neeman et al., 2022), and conflicts within an LLM's parametric knowledge (Elazar et al., 2021; Elsahar et al., 2018; Dong et al., 2023; Zhao et al., 2023), often stemming from noise, biases, or outdated information in the vast training corpora.

However, existing approaches (comprehensively discussed in Appendix A) face critical limitations when addressing temporal knowledge conflicts, a severely under-explored area of parametric knowledge conflicts. Current benchmarks rely heavily on popular entities from structured knowledge bases like Wikidata (Kim et al., 2023; Marjanović et al., 2024; Su et al., 2024b; Özer and Yıldız, 2025; Tang et al., 2025), leading to inadequate evaluation due to LLMs' strong memorization of such entities (Mallen et al., 2022). Moreover, the static nature of most benchmarks hinders fair evaluation of LLMs with differing knowledge cut-off dates (Tang et al., 2025). Other approaches employ Subject-Relation-Object extraction-based methods (Özer and Yıldız, 2025) that fail to capture nuanced con-

textual changes in complex corpora and often resort to artificially induced conflicts (Su et al., 2024b) rather than naturally arising temporal conflicts in real-world text. Hence, although temporal conflicts are among the most prevalent forms of knowledge inconsistency in practice (Kaddour et al., 2023; Zhou et al., 2023), our understanding of how LLMs handle naturally occurring temporal evolution remains severely limited.

To address these limitations, we propose a novel framework for systematic benchmark construction to evaluate LLMs' temporal knowledge handling from unstructured, time-stamped corpora (Section 2). Our framework identifies naturally occurring knowledge evolution through: (1) extracting salient entities and concepts, (2) clustering concepts to identify overarching topics, (3) isolating attributes that change over time, and (4) generating grounded questions with time-sensitive gold answers tailored to different LLM knowledge cut-off dates. Unlike previous studies relying on synthetic conflicts or structured knowledge bases, our approach directly applies to real-world data sources where information genuinely evolves, enabling authentic evaluation of temporal knowledge handling capabilities.

Using our framework, we introduce `evolveQA`, a comprehensive QA benchmark for evolving knowledge built across 3 dynamic domains: AWS service updates (550 QA pairs), Azure platform changes (203 QA pairs), and WHO-Disease Outbreak News reports (169 QA pairs), spanning multiple LLM knowledge cut-off dates and 3 question formats: open-ended, multiple-choice, and verifiable QA (Section 3). Our evaluation of 12 LLMs (6 open-source, 6 closed-source) from 4 model families reveals universal and significant performance degradation on temporally evolving knowledge, with accuracy drops of $6\% - 31\%$ across domains compared to non-conflicting scenarios. Critically, we show that question format substantially impacts performance: while models achieve $53\% - 76\%$ accuracy on MCQ questions about evolving facts, they drop to $12\% - 51\%$ on open-ended queries targeting the same knowledge. Interestingly, in $32\% - 45\%$ of cases, models generate outdated responses to open-ended questions but select the correct option in the corresponding MCQ format, indicating that updated knowledge exists in parametric memory but is poorly prioritized during recall (Section 4). Such findings provide crucial insights into temporal knowledge conflicts and establish a foundation for developing more temporally robust LLMs.

2 Temporally Evolving Knowledge Benchmark Creation Framework

In this section, we propose a novel framework to identify temporally evolving knowledge from a given text corpora and develop a benchmark of question-answer pairs to probe LLMs' understanding of this knowledge. This framework involves two major steps: (1) systematically navigating a corpus of time-stamped documents to *identify temporally evolving facts*; (2) *generating high-quality questions* that query LLMs for these evolving facts, along with their corresponding *gold-standard answers* tailored to different LLM knowledge cut-off dates. Our approach is outlined in Figure 2.

2.1 Identifying Temporally Evolving Facts

Our goal is to first identify evolving facts from a given text corpus where the most recent information is considered authoritative. To achieve this, we employ an LLM-based approach as follows.

2.1.1 Concept and Salient Entity Extraction

We begin with a text corpus tagged with publication timestamps. To identify the precise knowledge that evolves over time in this corpus, we first need to group documents discussing similar topics. However, real-world documents are inherently noisy, and often cover multiple sub-topics, making it difficult to cluster the documents into meaningful groups. To mitigate this, following previous works (Nakshatri et al., 2025; Tamkin et al., 2024), we rely on clustering abstract representations of the documents instead of the raw documents. In particular, we first distill documents into their key semantic components, which act as discriminative anchors that enable effective grouping of related documents. We define such a component as the combination of the following– (a) *concept:* a short text description that provides an abstract, yet informative view of a central discussion point in the document. (b) *entity*: salient subject associated with the identified concept.

For every document in the given time-stamped corpus, we use an LLM to extract up to k high-level *concepts* along with their corresponding *salient entities*. This pairing of an entity with its associated context provides a robust, topic-centric representation of the document's content (Chen et al., 2023; Nanni et al., 2017), forming a solid foundation for subsequent clustering. The *step-1* in Figure 2 shows an example of a salient entity: "AWS Snow-

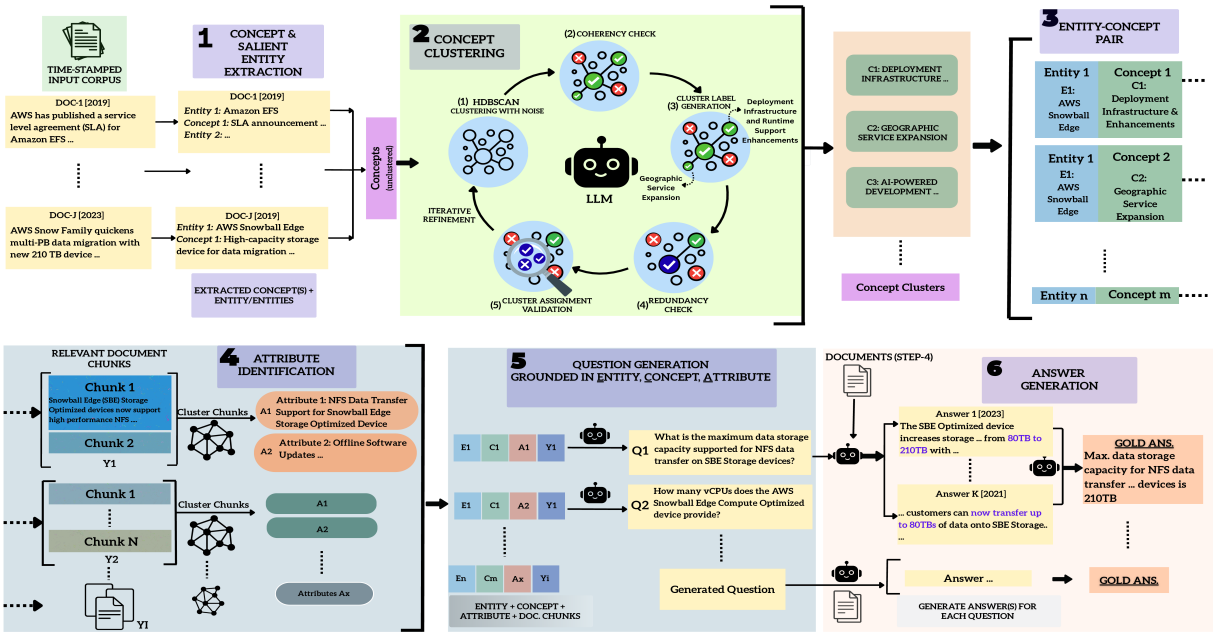

Figure 2: Framework for `evolveQA` construction. It operates in two phases: **1. Evolving Fact Identification:** (1) Extract entities and concepts, (2) cluster concepts into topics, (3) obtain documents for each `{Entity, Concept}` pair, (4) cluster document chunks to identify temporal attributes. **2. Grounded QA Curation:** (5) Generate specific questions and (6) curate time-sensitive gold answers for each `{Entity, Concept, Attribute}` tuple.

ball Edge", and a concept: "High-capacity storage device for data migration", extracted from a document in the AWS updates corpus.

2.1.2 High-Coherence Concept Clustering

In this step, we cluster the *concepts* extracted in the previous step rather than the raw documents. This approach enables embedding-based clustering algorithms to effectively identify semantically dense regions in the embedding space, leading to coherent initial topic groups. These topics are further refined iteratively with an LLM-in-the-loop approach. Unlike prior works that primarily use LLMs for cluster labeling and refinement (Nakshatri et al., 2023), or preference alignment (Zhang et al., 2023), our method leverages LLMs both (a) *during* the clustering process to label and refine the generated concept clusters, and (b) *post clustering* to validate and correct concept assignments. Our iterative process works as follows.

Step 1: Concept-Label Discovery. We begin by embedding all extracted concepts using a text embedding model and then apply HDBSCAN algorithm (Campello et al., 2013) to identify dense regions in the embedding space. Then we perform three critical LLM-guided refinement steps.

- **Coherency check:** To ensure high cluster coherence, we use an LLM to validate whether the top-m ($m = 5$) most central members within each cluster are semantically aligned with each other. If these members do not consistently discuss the same overall concept, the cluster is marked as *incoherent*, hence, dissolved and its constituent concepts are returned to an unclustered pool.
- **Cluster-label generation:** For each coherent cluster, an LLM is prompted to assign a concise label based on the semantic content of its top-m documents. This label serves as the overarching concept discussed in the cluster.
- **Redundancy check:** To avoid multiple clusters discussing the same concept, an LLM is used to validate if a cluster pair with semantically similar cluster labels (determined based on cosine similarity between cluster label embeddings) can be merged, ensuring conceptually distinct final clusters.

Step 2: Concept-Cluster Assignment Validation. In this step, we validate individual concept assignments. First, within each cluster, an LLM verifies if every concept is consistent with the assigned cluster label. Concepts deemed inconsistent are removed and returned to the unclustered pool. Subsequently, for any remaining unclustered concepts, an LLM attempts to assign them to one of the top-3 existing, cluster labels (obtained based on cosine similarity between the concept and cluster label embeddings). This process ensures the alignment of each concept in a cluster with its label.

Step 3: Iterative Refinement. If a significant number of concepts remain unclustered, we return to **Step 1** and repeat the process. This iterative

loop continues until all concepts are clustered or a maximum iteration count (set to 5) is reached. This iterative, LLM-guided process ensures that the final concept clusters are both highly coherent and conceptually distinct, laying a solid foundation for identifying evolving knowledge attributes.

2.1.3 Identifying Evolving Attributes

The generated concept cluster labels from the previous step are abstract in nature. For instance, "AI-Powered Development and Integration, Geographic Service Expansion etc." (shown as the output of *step-2* in Figure 2), are too abstract for generating precise questions about changing facts. One cannot ask a specific, answerable question based on such a broad theme alone. Therefore, in this step, by organizing the source corpora based on a {*entity, concept cluster label*} pair, we pinpoint the specific, fine-grained *attributes* within these documents that genuinely evolve over time. An *attribute* represents a specific property or detail, such as "Offline Software Updates for Air-Gapped Snowball Edge Devices", that is susceptible to temporal change.

To achieve this, we utilize the structure developed in the previous steps. For a concept cluster, we begin by aggregating all the (*entity, document*) pairs that correspond to the concepts within that cluster. This process naturally groups documents into topically focused subsets of the corpus, i.e, all documents related to an entity, say "AWS Snowball Edge", and a concept, say "Deployment Infrastructure and Enhancements", are grouped together.

Within this document subset, our goal is to isolate the discrete factual statements that undergo updates. Since documents may contain multiple attributes, we first segment them into smaller units (or chunks). We employ a sliding window of three sentences per chunk with one-sentence overlap to maintain contextual integrity. These chunks are then embedded and clustered using HBDSCAN algorithm. Each resulting cluster of chunks now represents a recurring, specific piece of information. An LLM is prompted to assign a descriptive label to each chunk cluster, which is referred to as the *attribute*. Formally, all document chunks within a single {*entity, concept cluster label, attribute*} tuple represent the same specific piece of potentially evolving knowledge.

This three-part tuple forms the fundamental unit for our question generation. For brevity in the subsequent sections, we will refer to the {*concept cluster label*} simply as the {*concept*}.

2.2 Question & Answer Generation

After identification of evolving knowledge *attributes*, here, we generate questions that target them and curate time-sensitive gold-standard answers for various LLM knowledge cut-off dates.

2.2.1 Question Generation

For question generation, we consider a specific cluster corresponding to a unique `(entity, concept, attribute)` tuple. To ensure questions are well-grounded, we sample the top-K chunks from different documents and from varying timestamps within this *attribute* cluster (based on cosine similarity to the cluster centroid). Using these selected chunks as contextual anchors, we prompt an LLM to formulate a question that is explicitly grounded in the three dimensions of `(entity, concept, attribute)` tuple. This grounding is crucial for ensuring precision and query disambiguation. The *entity*, say "AWS Snowball Edge", establishes the core subject; while the *attribute*, say "NFS Data Transfer Support", specifies the precise property or characteristic undergoing change. The inclusion of the *concept*, say "Deployment Infra. and Runtime Support Enhancements", is critical for contextual disambiguation, ensuring the question targets a specific facet of the entity rather than its broader context. This three-part hierarchical grounding directly leverages our structured knowledge representation to construct meaningful, targeted queries, preventing loosely framed questions that might elicit overly broad responses. An example question: "What is the maximum data storage capacity supported for NFS data transfer on Snowball Edge devices?", is shown in *step-5* of Figure 2.

To ensure high quality of our benchmark, we incorporate a post-generation cleaning step. We remove: (a) questions that elicit exhaustive lists as responses (e.g., "In which regions are EC2 R6i instances available?"). To correctly answer such questions, we would require a temporally complete *AWS* documentation coverage that is beyond the scope of our input corpus. (b) Any duplicate questions targeting semantically equivalent knowledge.

Knowledge Probing Formats: Previous work (Song et al., 2025) has shown that LLM performance significantly varies across different question types. To understand the impact of knowledge probing format in LLMs' ability to answer questions related to evolving knowledge, we generate three different question types as described below.

(1) Open-ended Questions: Obtained directly

using the question generation pipeline explained above. These questions evaluate the LLM's ability to extract the most up-to-date information from its parametric knowledge without any cues.

(2) Multiple-Choice Questions (MCQ): By presenting the correct answer alongside plausible distractors, we can assess an LLM's ability to identify current information when it is explicitly provided in context. For an open-ended question, we construct a corresponding MCQ variant using an LLM. Specifically, by providing the *open-ended question*, *gold answer* (detailed in Section 2.2.2) and a list of potential *distractor knowledge chunks* (from related concepts), we prompt an LLM to generate an MCQ question, which probes for the same knowledge as its open-ended counterpart. When generating distractors, we prioritize older and previously valid answers. If historical answers are insufficient in number, we generate plausible but incorrect distractors from related concepts.

(3) Verifiable QA: This format probes the consistency and robustness of an LLM's internal knowledge. For each open-ended question, we generate a pair of "yes/no" questions: (a) Yes-eliciting question: presents the correct and current fact and asks for confirmation (e.g., "Is *90 days* the maximum historical data retention period accessible through Amazon Connect's GetMetricDataV2 API?"). (b) No-eliciting question: presents a plausible but incorrect fact and asks for confirmation (e.g., "Is *14 days* the maximum historical data retention period accessible through Amazon Connect's GetMetricDataV2 API?")

2.2.2 Gold Answer Curation

For each question, all documents associated with the chunks within the *attribute* cluster are considered as potential sources for an answer. Our gold answer curation process employs a multi-stage, LLM-guided prompting strategy to ensure accuracy. First, for each potential source document, we prompt an LLM to determine if it contains sufficient and relevant evidence to answer the given question. This initial verification step serves as a critical guardrail, preventing the LLM from attempting to extract answers from irrelevant or insufficient textual segments. Next, based on the LLM's positive validation of evidence sufficiency, we instruct it to extract a precise, well-grounded answer from each identified relevant document. Finally, after obtaining individual answers from all relevant source documents, we use the LLM to synthesize and combine them into a comprehensive gold answer. This step ensures that the final gold answer accurately reflects the information across all available relevant documents for that specific question and timestamp.

Consideration of LLM Knowledge Cut-off Dates: Our gold answer curation process explicitly accounts for the knowledge cut-off date of the target LLM intended for evaluation. Specifically, during the first step of evidence verification, we filter the potential source documents, discarding any document whose timestamp exceeds the knowledge cut-off date of the LLM being evaluated. For instance, when constructing the benchmark for `Llama-3.1-8B-Instruct` model (whose knowledge cut-off date is December 2023), all documents published after this date are excluded from the answer curation process for that specific evaluation context. This ensures a fair and accurate evaluation of each LLM's inherent parametric knowledge. We perform a final cleaning of the generated QA pairs. We discard all non-evolving questions, for which only one single answer was identified across all timestamps before the target LLM's knowledge cut-off date. We also discard any questions for which the identified answers show no substantive knowledge change over time.

All the prompt structures used in different steps of our framework are shown in Appendix D.1.

3 Evolving Knowledge - Question Answering Benchmark (`evolveQA`)

Building upon the methodology described in Section 2, we construct `evolveQA`, a comprehensive QA benchmark specifically designed to test LLMs on temporally evolving knowledge. For its construction, we use `Claude Sonnet-3.5-v2` for all LLM calls. We acknowledge that, in principle, any sufficiently capable LLM can be substituted. All hyperparameters can be found in Appendix B.3.

3.1 Knowledge Domains

We curate `evolveQA` from 3 distinct, time-stamped corpora representing varying levels of structure and domain specificity. Their dynamic nature of knowledge makes them suitable for studying temporal knowledge conflicts. The domains are as follows.

AWS What's New Feed: This corpus comprises publicly available updates and announcements on Amazon Web Services (AWS) from 2019 to 2024.

Azure Updates: Similar in structure to the AWS feed, this corpus covers publicly available updates

on Microsoft Azure services. It provides a parallel, yet distinct, technical domain to test LLMs on evolving knowledge.

WHO - Disease Outbreak News (WHO-DONs): This corpus consists of articles published by the World Health Organization (WHO) regarding global disease and virus outbreaks, spanning from 1997 till 2025. Compared to cloud domains, WHO-DONs represents a significantly different domain. In addition, document length is at least ≈ 110 words longer than the other two domains with several sections. By breaking down each document into its respective sections, we ingest this structurally diverse data into our framework.

3.2 LLMs Used

As discussed in Section 2, an LLM's internal knowledge is temporally bounded by its training cut-off date, directly impacting how LLMs handle conflicting information without supporting mechanisms like RAG (Lewis et al., 2020). Consequently, we develop `evolveQA` for 12 prominent LLMs (6 open-source, 6 closed-source, spanning 4 model families) across 6 knowledge cut-off dates, with each LLM's evaluation set aligned to its respective cut-off date (collected from LLM providers' official websites).

To empirically validate LLMs lack knowledge beyond their reported cut-off dates, we perform Membership Inference Attack (MIA) (Shi et al., 2023) using `LLama-3.1-8B-instruct`. We find a consistent performance gap in each domain when comparing the relative scores, for the documents *before* and *after* the LLM's cut-off date. This confirms that the chosen input corpus and model cut-off dates are well-aligned with the model's internal knowledge (details in Appendix B.4).

3.3 Benchmark Quality

We employ safeguards to mitigate hallucinations and bias from `Claude Sonnet-3.5-v2` during question and answer curation. We adopt the self-correction strategy of Kim et al. (2023), which demonstrates that self-validation is effective in filtering out model-specific biases. Following this approach, during question generation, we provide the context: {*entity, concept, attribute*} (E-C-A) along with the set of relevant document chunks and prompt the LLM to validate if the knowledge in those chunks evolves with respect to the provided E-C-A, ensuring the questions remain grounded in E-C-A. Similarly, after answer curation, we prompt the LLM with a question and a list of all answers (potentially evolved with time) to validate the authenticity of temporal evolution. More information on these steps is provided in Appendix B.5.

Cutoff Dates	# of QA pairs	Relevant LLMs based on cutoff
08/31/23	717	Claude 3 Haiku (Anthropic, 2024a)
12/31/23	775	Llama 3.1 8B Inst., 3.3 70B Inst.; (Dubey et al., 2024)
04/30/24	818	Claude 3.5 Sonnet V2, V1 (Anthropic, 2024b)
05/31/24	849	GPT-4.1-mini (Achiam et al., 2023), GPT-5-mini[1]
07/31/24	886	Claude 3.5 Haiku (Anthropic, 2024b)
08/31/24	922	Gemma3 4B, 27B Inst. (Team et al., 2025); Llama4–Scout, Maverick Inst. (Meta, 2025)

Table 1: `evolveQA` statistics for different cutoff dates. Full data distribution for each domain is provided in Appendix B.1.

We further analyze the quality of `evolveQA` through a human evaluation on a randomly sampled 100 QA pairs (see Appendix B.6.1 for more details). We observe that the questions from our benchmark are well-grounded to their {*entity, concept, attribute*} tuples in 92% of the cases, 94% of the identified attributes evolved over time, and 89% of the questions are of high quality−targeting specific non-trivial knowledge. On coupling this with the gold answer correctness of $> 91\%$, we ensure the reliability of our benchmark for LLM evaluation. Table 1 shows `evolveQA` statistics.

4 Evaluation of LLMs on `evolveQA`

4.1 Experimental Setup

Control Dataset: To benchmark LLMs' performance on evolving knowledge, we first establish their baseline performance in each domain using a control dataset. We construct a temporally non-conflicting control set by removing the documents from source corpora that are associated with top-4 most temporally-conflicting concept-clusters (identified during `evolveQA` creation), then excluding all documents pertaining to questions in `evolveQA`, sampling ≈ 1200 remaining documents, and generating QA pairs targeting static facts. This control dataset enables us to isolate the impact of temporal conflicts and establish each model's baseline performance on static facts within the same domains (see Appendix B.2 for control set statistics).

Evaluation Metrics: We evaluate LLMs' performance on `evolveQA` and control datasets using two metrics: **(a) Accuracy:** The proportion of correct answers (measured for all three question types). For open-ended QA, we also measure Outdated

[1] https://platform.openai.com/docs/models/gpt-5-mini

Response Rate (ORR), the proportion of answers that are factually correct for a past time period but are outdated by current knowledge. **(b) Hard Accuracy for Verifiable QA:** The proportion of cases where a model correctly answers both yes and no-eliciting variants of the same underlying fact.

Evaluation Methodology: Since exact-match metrics are insufficient for open-ended QA evaluation, we employ **model-as-a-judge** evaluation (Zheng et al., 2023) using `Claude Sonnet-4`, chosen for its strong instruction-following and semantic understanding capabilities. We provide the judge with the *question*, *LLM-generated response*, *gold answer*, and *answers* from previous timestamps. Then we prompt it to classify responses into three categories: *CORRECT_AND_CURRENT:* the response matches the current authoritative answer, *OUTDATED:* the response aligns with a previously valid but now outdated answer, or *INCORRECT:* the response is neither current nor historically valid. The evaluation rubric prompt is in Appendix D.3. We also perform a meta evaluation of the quality of the LLM judge and observe that the judge achieves an accuracy of $\approx 96\%$ while evaluating open-ended responses (details in Appendix B.6.2).

4.2 Results and Ablations

Main Results: The main results are presented in Table 2. We observe a consistent and significant degradation in performance for all evaluated models on the temporally conflicting `evolveQA` benchmark compared to the non-conflicting control dataset. In open-ended QA, we observe the most substantial and consistent performance drops across all domains when compared to the performance in the control set. In *AWS domain*, we observe performance drops ranging between $\approx 6\% - 26\%$. It is even more severe in the *WHO-DONs* domain, with reductions between $\approx 8\% - 31\%$, while it is up to 27% in *Azure* domain, when measured across all models. This confirms that even state-of-the-art LLMs struggle to retrieve and prioritize the most current information when faced with evolving facts stored in their parameters.

Impact of Question Format: We observe that performance varies across question types. Accuracy improves when the question is framed as an MCQ question or a verifiable question, when compared to open-ended question format (Table 2). Specifically, we observe accuracy improvements of $\approx 22\% - 54\%$ in MCQ format and $\approx 10\% - 30\%$ in verifiable QA format, when compared against the open-ended questions in `evolveQA`. It indicates that presence of the correct information in the context either in MCQ format or in the verifiable QA format, helps the LLMs select the correct answer. However, in both of the question types, performance lags significantly when compared to the control dataset. This finding also verifies that the LLMs have the most up-to-date information in their parametric knowledge, however, fails to recall them when probed in an open-ended setting.

Notable Model-Specific Anomalies: In the Azure domain, for a few LLMs (e.g., Claude Haiku, Llama 3), MCQ performance on `evolveQA` is comparable or even slightly higher than the performance in the control dataset. Upon deep dive, we identify 17 instances in the non-conflicting control dataset where `Claude-3.5-Haiku` answered correctly in the open-ended setting but failed in its corresponding MCQ format (examples provided in Appendix B.7.1), which contributed to its lower accuracy in the control dataset. Surprisingly, we do not observe this behavior on `evolveQA` with `Haiku`. With `Llama-3.1-8B model`, we observe a comparatively minor gap in performance between the control dataset and `evolveQA`. This can be attributed to its lower performance in general, as seen in open-ended QA format for *Azure* and *AWS Domains*. In verifiable QA control set, we observe that `Gemma-4B` is generally biased to responding "yes", agreeing with $\approx 81\%$ of "yes"-eliciting questions but also incorrectly agreeing with $\approx 63\%$ of "no"-eliciting questions (averaged across the 3 domains). Since `Gemma-4B` fails to correctly answer both the "yes-no" questions, its *Hard Acc.* performance is lower across the 3 domains in the control dataset.

Outdated Response Rate (ORR): LLMs provide outdated responses more frequently than incorrect ones in open-ended QA, with `ORR` ranging $\approx 23\% - 54\%$ (Appendix C.1). Interestingly, correct knowledge is often latently present even when outdated responses are generated. We test this by examining cases where a model provided an outdated answer in open-ended format, however, selected the correct option in the corresponding MCQ format. This occurred in 32.44% of cases in *WHO-DONs*, 41.34% in *Azure*, and 45.2% in *AWS*: indicating that models possess updated knowledge but fail to recall it during open-ended generation.

Impact of Explicit Context in Prompt: We investigate whether providing more context related to the question helps LLMs handle temporal conflicts in open-ended QA by testing three prompt varia-

Domains	Model Families	Models	Open-Ended QA (%)		MCQ (%)		Verifiable QA (%)	
			evolveQA Acc.	Control Acc.	evolveQA Acc.	Control Acc.	evolveQA Hard Acc.	Control Hard Acc.
AWS Whatsnew Feed	Anthropic	Claude 3.5-sonnet-v2	26.95	46.39	63.33	82.49	42.64	68.03
		Claude 3.5-sonnet-v1	22.9	44.41	63.33	83.47	40.94	67.05
		Claude 3 Haiku	15.93	33.84	62.47	68.07	36.99	68.39
		Claude 3.5 Haiku	22.64	42.14	67.69	73.24	37.82	43.89
	Llama	Llama 3.1-8B-Instruct	14.44	20.33	61.84	67.53	35.76	49.04
		Llama 3.3-70B-Instruct	18.53	33.13	64.99	73.78	36.64	72.7
		Llama 4-Scout-instruct	17.09	31.78	68.32	75.35	41.92	65.43
		Llama 4-Maverick-instruct	22.18	36.45	66.3	79.51	40.04	80.53
	OpenAI	GPT-4.1-mini	22.86	41.69	62.78	78.24	39.84	60
		GPT-5-mini	29.62	55.4	59.96	82.08	42.71	65.71
	Gemma	Gemma3-4B-instruct	12.18	18.83	52.85	58.24	35.53	25.62
		Gemma3-27B-instruct	21.82	34.71	64.46	71.65	39.29	47.07
Azure Updates	Anthropic	Claude 3.5-sonnet-v2	22.95	49.82	70.33	77.73	49.42	58.33
		Claude 3.5-sonnet-v1	21.31	46.24	70.88	78.91	47.09	55.95
		Claude 3 Haiku	18.18	27.1	70.78	64.44	40.41	58.72
		Claude 3.5 Haiku	21.32	41.5	68.72	64.75	40.22	39.69
	Llama	Llama 3.1-8B-Instruct	17.58	16.85	63.64	60.4	45.81	43.09
		Llama 3.3-70B-Instruct	24.24	36.26	70.3	73.2	46.45	59.35
		Llama 4-Scout-instruct	18.72	31.58	72.64	71.76	48.4	55.04
		Llama 4-Maverick-instruct	23.15	37.19	76.12	75.95	47.34	75.97
	OpenAI	GPT-4.1-mini	26.84	38.35	71.81	76.56	39.89	55.95
		GPT-5-mini	29.47	49.82	67.55	76.95	42.13	51.19
	Gemma	Gemma3-4B-instruct	14.29	15.09	54.73	51.53	32.98	29.46
		Gemma3-27B-instruct	25.12	30.88	69.15	64.12	46.28	40.7
WHO—Disease Outbreak News	Anthropic	Claude 3.5-sonnet-v2	38.41	64.09	73.1	86.19	67.67	79.31
		Claude 3.5-sonnet-v1	34.23	62.43	71.72	90.61	63.91	76.44
		Claude 3 Haiku	29.41	60.69	66.67	81.5	47.9	81.93
		Claude 3.5 Haiku	33.33	63.44	72.08	80.11	58.62	76.97
	Llama	Llama 3.1-8B-Instruct	27.4	45.71	67.36	73.14	37.5	41.67
		Llama 3.3-70B-Instruct	39.04	59.43	68.75	83.43	53.12	72.62
		Llama 4-Scout-instruct	31.36	55.91	68.71	82.26	56.67	75.28
		Llama 4-Maverick-instruct	33.14	59.14	76.07	84.95	54.19	86.45
	OpenAI	GPT-4.1-mini	39.1	68.13	72.19	81.87	62.14	81.71
		GPT-5-mini	50.64	71.43	72.85	86.26	67.86	86.29
	Gemma	Gemma3-4B-instruct	23.67	31.72	63.19	64.52	41.94	19.1
		Gemma3-27B-instruct	39.05	63.98	71.17	79.51	50.97	61.24

Table 2: Performance comparison of 4 model families across 3 data domains. Results are reported for 3 different knowledge probing formats− for both `evolveQA` and non-conflicting control dataset. Results are based on a single execution of the LLM.

tions: (1) providing ground-truth Entity-Concept-Attribute (E-C-A) from our benchmark in prompt; (2) providing "current date" in the prompt; (3) providing both E-C-A and "current date". Results show that providing only the "current date" generally does not improve LLMs ability to recall authoritative information. However, providing E-C-A context generally improves performance, providing gains up to 11% (see Appendix C.3 for details).

Impact of Superseding Knowledge: Conflicts in `evolveQA` fall into two types: (1) *additive knowledge* complementing existing facts (e.g., "How does Kendra access user information?" has multiple valid answers); (2) *superseding knowledge* invalidating previous facts (e.g., "How many AWS regions support Kendra?" has only one valid answer). To analyze more challenging superseding knowledge, we isolate MCQ instances where outdated facts serve as distractors. Models show increased vulnerability: MCQ accuracy drops $\sim 15\%$ (`GPT-4.1-mini` in *AWS*/*WHO-DONs*, `GPT-5-mini` in *Azure*), suggesting strong interference from historically correct distractors. Open-ended QA performance drops up to 9%. Notably, GPT family models show consistently higher degradation across both formats, indicating potential architectural sensitivity to factual updates (Appendix C.2).

Impact of Model Judge Bias: To ensure our open-ended QA evaluation using `Claude Sonnet 4` is not skewed by self-preference bias, we perform cross-validation using `GPT-4.1-mini` to re-evaluate all responses from the *Claude* family. Results in Table 11 (Appendix) confirm robustness: while absolute scores vary between judges, performance trends and degradation magnitude on `evolveQA` remain consistent.

5 Conclusion

In this paper, we introduce `evolveQA` to evaluate LLMs on temporal knowledge conflicts using naturally evolving real-world data. Our findings reveal that while LLMs contain updated knowledge in their parameters, they struggle to retrieve it reliably, particularly in open-ended queries. This work exposes critical limitations of modern LLMs and provides a benchmark for developing temporally robust LLMs in dynamically evolving domains.

Limitations

Availability of temporal information: Our methodology relies on availability of reliable, timestamped documents. The entire pipeline, from identifying evolving facts to curating time-sensitive gold answers, relies on this metadata. Consequently, our framework cannot be directly applied to corpora lacking explicit chronological information.

Limitation in quantifying Recall: A key methodological challenge is quantifying the recall of our fact identification process. The absence of a comprehensive ground-truth dataset for evolving facts within unstructured corpora is an inherent limitation of this task. Consequently, while we can evaluate the precision of the facts we identify, we cannot measure the completeness of our framework in capturing every possible instance of evolving knowledge. For a similar reason, during the construction of the control dataset, we cannot quantify the number of cases where we successfully identify static knowledge.

Modules and languages studied: Our approach is limited to the text module in the English language only, while evolving knowledge in other modules (e.g., image, table, and so on) and languages can also be explored. We leave it as a future work.

Dependency on backbone LLM's capability: Our approach depends on a number of LLM calls in order to leverage their reasoning capability in different stages of the proposed algorithm. In our experiment, we use `Sonnet-3.5` as the backbone LLM, however, we note that any other reasonably capable LLM can be used for the prompting steps. We leave the correlation study between the quality of the resulting benchmark and the capability of the backbone LLM, as a future work.

Fixed cost for benchmark construction: To ensure high quality of our benchmark, our LLM-based approach uses `Claude Sonnet 3.5` throughout the pipeline. This results in a fixed cost of roughly 400 USD for constructing `evolveQA`. We note that this is a lot cheaper than relying on human annotations to evaluate LLMs on temporally conflicting data from three domains.

Knowledge conflict mitigation: Our study focuses on understanding how LLMs handle conflicting information stored in their parametric knowledge. While some studies have suggested using retrieval-augmented generation (RAG) to address these conflicts, our goal is different. Instead of trying to fix these conflicts, we aim to understand how LLMs behave when they encounter contradictory information. We hope this understanding will help develop solutions during model training or through other post-training methods in the future.

Limited temporal scope of the input corpus: Our study relies on datasets with limited temporal scope. For instance, *WHO-DONs* covers updates from 1997−2024. As a result, our experiments are restricted to question types for which we have sufficient evidence within this period. In particular, we cannot have questions requiring *exhaustive* or *comprehensive* coverage. Questions such as "In which regions did HCV outbreak occur?" cannot be fully answered or automatically evaluated without access to a temporally *complete* source documentation. This limitation stems from the unavailability of complete knowledge in the domains in the corresponding websites.

Ethical Considerations

To the best of our knowledge, we did not violate any ethical code while conducting the research work described in this paper. We reported all implementation and dataset details in the paper for reproducibility and upon acceptance we will release our codes and datasets for future research. The datasets (from public websites) used in this paper are publicly available and permitted for scientific research. The human evaluation details are presented in Appendix B.6. We perform extensive ablation studies and meta evaluation of the model judge used in the work, in order to provide the readers an idea about potential error patterns and risks associated with our proposed method and reported results.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI Anthropic. 2024a. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1(1):4.

AI Anthropic. 2024b. Claude 3.5 sonnet model card addendum. *Claude-3.5 Model Card*, 3(6).

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45.

Hung-Ting Chen, Michael JQ Zhang, and Eunsol Choi. 2022. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. *arXiv preprint arXiv:2210.13701*.

Sihao Chen, William Bruno, and Dan Roth. 2023. Towards corpus-scale discovery of selection biases in news coverage: Comparing what sources say about entities as a start. *arXiv preprint arXiv:2304.03414*.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. *ArXiv*, abs/2108.06314.

Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Qingxiu Dong, Jingjing Xu, Lingpeng Kong, Zhifang Sui, and Lei Li. 2023. Statistical knowledge assessment for large language models. *Advances in Neural Information Processing Systems*, 36:29812–29830.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Bahare Fatemi, Mehran Kazemi, Anton Tsitsulin, Karishma Malkan, Jinyeong Yim, John Palowitch, Sungyong Seo, Jonathan Halcrow, and Bryan Perozzi. 2024. Test of time: A benchmark for evaluating llms on temporal reasoning. *arXiv preprint arXiv:2406.09170*.

Vivek Gupta, Pranshu Kandoi, Mahek Vora, Shuo Zhang, Yujie He, Ridho Reinanda, and Vivek Srikumar. 2023. TempTabQA: Temporal question answering for semi-structured tables. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2431–2453, Singapore. Association for Computational Linguistics.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. 2022a. TemporalWiki: A lifelong benchmark for training and evaluating ever-evolving language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6237–6250, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. 2022b. Towards continual knowledge learning of language models. *Preprint*, arXiv:2110.03215.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*.

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A. Smith, Yejin Choi, and Kentaro Inui. 2024. Realtime qa: What's the answer right now? *Preprint*, arXiv:2207.13332.

Yujin Kim, Jaehong Yoon, Seonghyeon Ye, Sangmin Bae, Namgyu Ho, Sung Ju Hwang, and Se-Young Yun. 2023. Carpe diem: On the evaluation of world knowledge in lifelong language models. *arXiv preprint arXiv:2311.08106*.

Evgenii Kortukov, Alexander Rubinstein, Elisa Nguyen, and Seong Joon Oh. 2024. Studying large language model behaviors under context-memory conflicts with real documents. *arXiv preprint arXiv:2404.16032*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. *Preprint*, arXiv:2205.11388.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2022.

When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Sara Vera Marjanović, Haeun Yu, Pepa Atanasova, Maria Maistro, Christina Lioma, and Isabelle Augenstein. 2024. Dynamicqa: Tracing internal knowledge conflicts in language models. *arXiv preprint arXiv:2407.17023*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

AI Meta. 2025. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. *https://ai. meta. com/blog/llama-4-multimodal-intelligence/, checked on*, 4(7):2025.

Seyed Mahed Mousavi, Simone Alghisi, and Giuseppe Riccardi. 2024. DyKnow: Dynamically verifying time-sensitive factual knowledge in LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8014–8029, Miami, Florida, USA. Association for Computational Linguistics.

Nishanth Nakshatri, Siyi Liu, Sihao Chen, Dan Roth, Dan Goldwasser, and Daniel Hopkins. 2023. Using llm for improving key event discovery: Temporal-guided news stream clustering with event summaries. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4162–4173.

Nishanth Nakshatri, Nikhil Mehta, Siyi Liu, Sihao Chen, Daniel J Hopkins, Dan Roth, and Dan Goldwasser. 2025. Talking point based ideological discourse analysis in news events. *arXiv preprint arXiv:2504.07400*.

Federico Nanni, Simone Paolo Ponzetto, and Laura Dietz. 2017. Building entity-centric event collections. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 1–10. IEEE.

Ella Neeman, Roee Aharoni, Or Honovich, Leshem Choshen, Idan Szpektor, and Omri Abend. 2022. Disentqa: Disentangling parametric and contextual knowledge with counterfactual question answering. *arXiv preprint arXiv:2211.05655*.

Atahan Özer and Çağatay Yıldız. 2025. Question answering under temporal conflict: Evaluating and organizing evolving knowledge with llms. *arXiv preprint arXiv:2506.07270*.

Liangming Pan, Wenhu Chen, Min-Yen Kan, and William Yang Wang. 2021. Attacking open-domain question answering by injecting misinformation. *arXiv preprint arXiv:2110.07803*.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*.

Seok Hwan Song, Mohna Chakraborty, Qi Li, and Wallapak Tavanapong. 2025. Is large language model performance on reasoning tasks impacted by different ways questions are asked? *arXiv preprint arXiv:2507.15707*.

Zhaochen Su, Juntao Li, Jun Zhang, Tong Zhu, Xiaoye Qu, Pan Zhou, Yan Bowen, Yu Cheng, and 1 others. 2024a. Living in the moment: Can large language models grasp co-temporal reasoning? *arXiv preprint arXiv:2406.09072*.

Zhaochen Su, Jun Zhang, Xiaoye Qu, Tong Zhu, Yanshu Li, Jiashuo Sun, Juntao Li, Min Zhang, and Yu Cheng. 2024b. Conflictbank: A benchmark for evaluating the influence of knowledge conflicts in llm. *arXiv preprint arXiv:2408.12076*.

Alex Tamkin, Miles McCain, Kunal Handa, Esin Durmus, Liane Lovitt, Ankur Rathi, Saffron Huang, Alfred Mountfield, Jerry Hong, Stuart Ritchie, and 1 others. 2024. Clio: Privacy-preserving insights into real-world ai use. *arXiv preprint arXiv:2412.13678*.

Wang-Chiew Tan, Jane Dwivedi-Yu, Yuliang Li, Lambert Mathias, Marzieh Saeidi, Jing Nathan Yan, and Alon Y Halevy. 2023. Timelineqa: A benchmark for question answering over timelines. *arXiv preprint arXiv:2306.01069*.

Wei Tang, Yixin Cao, Yang Deng, Jiahao Ying, Bo Wang, Yizhe Yang, Yuyue Zhao, Qi Zhang, Xuanjing Huang, Yu-Gang Jiang, and Yong Liao. 2025. EvoWiki: Evaluating LLMs on evolving knowledge. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 948–964, Vienna, Austria. Association for Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

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. Fresh-LLMs: 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.

Fei Wang, Xingchen Wan, Ruoxi Sun, Jiefeng Chen, and Sercan Ö. Arık. 2025. Astute rag: Overcoming imperfect retrieval augmentation and knowledge conflicts for large language models. *https://arxiv.org/abs/2410.07176*.

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2023. Resolving knowledge conflicts in large language models. *arXiv preprint arXiv:2310.00935*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Yifan Wei, Yisong Su, Huanhuan Ma, Xiaoyan Yu, Fangyu Lei, Yuanzhe Zhang, Jun Zhao, and Kang Liu. 2023. MenatQA: A new dataset for testing the temporal comprehension and reasoning abilities of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1434–1447, Singapore. Association for Computational Linguistics.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. 2025. Livebench: A challenging, contamination-limited llm benchmark. *Preprint*, arXiv:2406.19314.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2023. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations*.

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuwei Zhang, Zihan Wang, and Jingbo Shang. 2023. Clusterllm: Large language models as a guide for text clustering. *arXiv preprint arXiv:2305.14871*.

Yukun Zhao, Lingyong Yan, Weiwei Sun, Guoliang Xing, Chong Meng, Shuaiqiang Wang, Zhicong Cheng, Zhaochun Ren, and Dawei Yin. 2023. Knowing what llms do not know: A simple yet effective self-detection method. *arXiv preprint arXiv:2310.17918*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021.

A Related Work

Knowledge Conflict in LLMs. Understanding the effect of different types of knowledge conflicts in LLMs has recently gained much interest. For example, conflicts in retrieved context (Pan et al., 2021; Chen et al., 2022; Wang et al., 2025), conflicts between external context and LLMs' internal parametric knowledge (Neeman et al., 2022; Wang et al., 2023; Xie et al., 2023; Kortukov et al., 2024), and intra-memory conflicts within an LLM's parameters (Elsahar et al., 2018; Elazar et al., 2021; Dong et al., 2023; Zhao et al., 2023; Marjanović et al., 2024). Our work addresses this third category, focusing specifically on temporal knowledge conflicts that arise from outdated information in pre-training corpora (Kaddour et al., 2023; Zhou et al., 2023).

Evolving knowledge benchmarks. Most existing benchmarks for evolving or conflicting knowledge rely on structured knowledge bases, predominantly Wikidata (Kim et al., 2023; Su et al., 2024b; Marjanović et al., 2024; Tang et al., 2025; Özer and Yıldız, 2025). These works leverage versioned snapshots or change logs to identify evolving facts, typically represented as structured triples. While effective in controlled settings, such benchmarks depend on explicit temporal metadata that is rarely available in real-world corpora and remain limited to few structured knowledge bases. Some benchmarks additionally induce temporal conflicts artificially, e.g., by modifying timestamps or factual values (Su et al., 2024b), which does not reflect naturally occurring knowledge evolution.

Another line of work studies dynamic nature of knowledge and its implications on LLMs, including CKL-LAMA (Jang et al., 2022b) and TemporalWiki (Jang et al., 2022a), which assesses knowledge retention and updates. RealtimeQA-styled benchmarks (Kasai et al., 2024; White et al., 2025), and Dyknow (Mousavi et al., 2024) enables testing LLMs on most up-to-date knowledge, rather than whether their parametric knowledge conflicts with newer evidence. Moreover, most benchmarks do not explicitly account for model knowledge cut-off dates, with EvoWiki (Tang et al., 2025) being a notable exception, though it remains Wikidata-based.

Finally, human-curated benchmarks such as SituatedQA, StreamingQA, and FreshQA (Zhang and Choi, 2021; Liška et al., 2022; Vu et al., 2024) capture temporal evolution via manual question writing or annotation. While more natural, these benchmarks are costly to scale and typically static once released. In contrast, evolveQA is constructed fully automatically from unstructured, time-stamped corpora, does not rely on structured knowledge bases

or artificial conflicts, and explicitly models temporal knowledge conflicts arising from LLM knowledge cut-off dates, enabling evaluation in realistic and domain-specific settings.

Temporal Reasoning. We distinguish our focus on temporal knowledge conflicts from the broader field of temporal reasoning where the focus is on testing LLMs' ability to understand basic temporal concepts such as ordering events chronologically, determining how long events last, and understanding temporal relationships among events (Chen et al., 2021; Dhingra et al., 2022; Gupta et al., 2023; Wei et al., 2023; Tan et al., 2023; Fatemi et al., 2024). In contrast, our work focuses on the (in)consistency of factual knowledge stored within an LLM's parameters, as that knowledge evolves over time.

B evolveQA

B.1 Data Distribution - Across domains and Knowledge Probing Formats

Table 3 provides the complete distribution of evolveQA across the three domains of interest– {*AWS Feed, Azure, WHO-Disease Outbreak News (WHO-DONs)*}; It also provides the distribution for each knowledge probing format along with the relevant LLMs associated with each knowledge cut-off date.

B.2 Control Set - Data Distribution

Table 4 shows the detailed statistics for each knowledge cut-off date and across each domain under consideration, which is kept consistent with evolveQA for a fair comparison.

B.3 Hyperparameters / Implementation Details

Obtaining Domain Data for evolveQA: We scrape domain-specific data using Selenium[2]– Python package, from the following websites for building evolveQA. For constructing questions from *Azure* domain, we scraped data from https://azure.microsoft.com/en-us/updates website. For *AWS Whatsnew Feed*, we scraped the data from https://aws.amazon.com/new/. We note that several links from this website were missing, and as a result, we relied on Wayback Machine [3] for obtaining the data. Finally, for *WHO-DONs* domain, we scraped the articles from https://www.who.int/emergencies/disease-outbreak-news.

Curating evolveQA: (1) *Number of concepts k:* To determine k during concept extraction, we performed a qualitative study over a randomly sampled set of 10 documents from each domain. We observed that the documents in cloud domain did not discuss as many different elements as WHO-DONs domain. Therefore, based on the domain and average document length in each domain, we set $k = 3$ for *AWS Feed*, and *Azure* domains; we set $k = 5$ for *WHO-DONs* domain.

(2) *Additional Pre-processing for WHO-DONs domain:* When processing documents from WHO-DONs domain, we consider only the *virus* entities and all their associated documents for question generation and answer curation. We follow this pre-processing step to ensure that the data for different viruses do not mix with each other. For instance, if we consider a non-virus salient entity such as "WHO" and a concept "Prevention Measures for Disease Outbreaks", aggregating all documents associated with this entity-concept pair would imply that guidelines for different types of viruses would be aggregated with each other. This does not help with obtaining questions targeting specific piece of knowledge. Therefore, to discard such questions, we focus solely on *virus-specific* entities from this domain.

(3) *Entity Normalization:* Before organizing the corpus based on an entity-concept pair, we also need to ensure that all the entities are in its canonical form. This way, we can potentially reduce errors during the gold answer curation phase, resulting from highly relevant but missing time-sensitive documents. To achieve this, we let the LLM come up with a canonical name during the *concept and salient entity extraction* phase, as described in the prompt in Figure 5. Additionally, we also cluster the entities to ensure that all mentions of the same entity are grouped together for the subsequent stages of our pipeline.

(4) *Other Hyperparameters:* We use cohere-english[4] for embedding all the textual components in our framework. Prior to clustering the textual embeddings with HDBSCAN algorithm, to mitigate the issues with clustering sparse vectors, we reduce their dimensions using UMAP (McInnes et al., 2018) algorithm

[2] https://selenium-python.readthedocs.io/
[3] https://web.archive.org
[4] https://docs.cohere.com/docs/models

Cutoff Date	No. of QA Pairs Across Different Knowledge Probing Formats (evolveQA)									Total No. of Open-Ended QA Pairs	Relevant LLMs based on cutoff
	AWS Feed			Azure Updates			WHO-DONs				
	Open-ended	MCQ	Ver. QA	Open-ended	MCQ	Ver. QA	Open-ended	MCQ	Ver. QA		
8/31/23	427	421	419	154	154	146	136	132	119	717	Claude 3 Haiku
12/31/23	464	457	453	165	165	155	146	144	128	775	Llama 3.1-8B-inst, 3.3-70B inst.
4/30/24	486	480	469	183	182	172	149	145	133	818	Claude 3.5 Sonnet V2, V1
5/31/24	503	497	487	190	188	178	156	151	140	849	GPT-4.1-mini, GPT-5-mini
7/31/24	530	523	513	197	195	180	159	154	145	886	Claude 3.5 Haiku
8/31/24	550	543	532	203	201	188	169	163	155	922	Gemma3 4B, 27B inst, Llama4-Scout, Maverick inst.

Table 3: Shows a detailed breakdown of `evolveQA` (open-ended question format) across the three domains - {*AWS, Azure, and WHO-DONs*} domain. The table also provides the breakdown with respect to different knowledge probing formats across the three domains. Along with these, it shows the relevant LLMs intended for evaluation for each knowledge cut-off date.

Cutoff Date	No. of QA pairs			Total No. QA Pairs
	AWS Feed	Azure Updates	WHO-DONs	
8/31/23	597	262	173	1032
12/31/23	664	273	175	1112
4/30/24	707	279	181	1167
5/31/24	722	279	182	1183
7/31/24	738	284	186	1209
8/31/24	749	285	186	1220

Table 4: Shows the data distribution for the control set across the three domains for each knowledge cut-off date (consistent with `evolveQA` for a fair comparison.

Domain	Min-K% Prob	
	% Members (Before Cutoff)	% Non-Members (After Cutoff)
Azure	81	70.6
AWS	76.94	50.38
WHO	81.8	63.29

Table 5: Validation of pre-training exposure via Membership Inference Attack (MIA). Using the `min-k%` algorithm ($k = 0.2$), `Llama-3.1-8B-Instruct` was tasked to differentiate between 500 randomly sampled member (pre-cutoff) and 500 non-member (post-cutoff) documents. The resulting 11% to 26% point gap in detection rates between members and non-members across all domains confirms that our benchmark targets knowledge represented within the model's parameters.

($n_neighbors = 15, umap_components = 5$). For HDBSCAN clustering, we use the following hyperparameters: $min_cluster_size = 5; min_samples = 1; epsilon = 0.1$. Finally, for all LLM-calls, we set $temperature = 0$.

Evaluating LLMs: For analyzing the behavior of LLMs on `evolveQA`, we set $temperature = 0, top_p = 0.95$ for all models except `Llama-3.1-8B-instruct`, and `GPT-5-mini`. By setting $temperature = 0$, we faced token repetition issues with `Llama-3.1-8B-instruct`, and therefore, for this model, we set $temperature = 0.4$ for all our experiments. With `GPT-5-mini`, the model provider does not allow us to modify temperature parameter and all our experiments with this model is with the default setting (as provided by OpenAI).

B.4 Membership Inference Attack

Table 5 shows the results from MIA using *min-k%* algorithm. We conducted this experiment with `Llama-3.1-8B-Instruct` for $k = 0.2$. First, for each domain, we randomly sampled 500 data points (500 from *before* the model cut-off period, and 500 from after the model cut-off period). We applied *min-k%* algorithm on both these data samples and observed a significant gap in the detection rates across these two sets, indicating that models do not possess knowledge beyond their cut-off period.

B.5 `evolveQA` construction details

In this section, we provide details involved in constructing `evolveQA`, especially regarding the safeguards employed to prevent potential hallucinations and bias from the backbone LLM `Claude Sonnet-3.5-v2`. We employ safeguards during the question generation and after answer curation, to filter out all the questions that indicate no knowledge evolution, following (Kim et al., 2023). Specifically, during question generation, using the prompt shown in Figure 10, by providing the {*entity, concept, attribute*} (E-C-A) context along with the relevant document chunks, we ask an LLM to validate if the knowledge contained in the document chunks evolve with respect to the provided E-C-A context. Similarly, after the answer curation, for a question-answer pair across all knowledge cut-off dates, by providing the question along with a list of all potentially temporally evolving answers, we prompt the LLM to validate if the answer really evolves over time (using the prompt 16). If not, we discard such question-answer pairs. Along with these, we also perform other forms of filtering to clean up the data (discussed in Section 2.2.1).

B.6 Human Evaluation

This section discusses results from our human evaluation: (1) human assessment on the quality of the benchmark (`evolveQA`); (2) meta evaluation of our model judge (`Claude Sonnet 4`).

B.6.1 Evaluating evolveQA: Evidence from Human Assessment

To assess the quality of evolveQA, we conducted a human evaluation. For this evaluation, we sampled 100 question-answer pairs considering the validation of the LLM-judge. We used the same sample to perform a meta evaluation of the model judge (Section B.6.2). In particular, we randomly sampled 50 instances from the cases where the LLM-judge provided a *CORRECT_AND_CURRENT* label as the judgement, 30 randomly sampled from the cases where the LLM-judge provided *INCORRECT* as the judgement, 20 randomly sampled from the cases where the LLM-judge provided *OUTDATED* label as the judgement; across different domains and different model cutoffs. Four annotators (among the authors of this paper) were tasked with evaluating two key components: the generated questions and the curated gold answers.

For **Question Evaluation**, we assessed three dimensions: (a) **Groundedness:** measures if the question is faithfully grounded in its corresponding entity, concept, attribute (E-C-A) tuple. (b) **Knowledge Evolution:** validates if the attribute targeted by the question genuinely evolves over time, ensuring the question is relevant to our study of temporal knowledge conflicts. (c) **Quality & Specificity:** evaluates if the question targets non-trivial, specific information, avoiding generic or overly simple queries.

For **Gold Answer Correctness**, annotators verified the factual correctness of the curated answers by cross-referencing them against both the provided top-7 documents from source corpora and external search engine results.

Each datapoint was annotated by one annotator. Table 6 shows the results from human evaluation. It confirms the high quality of our generation pipeline. Specifically, the questions from our benchmark demonstrate high performance on individual quality dimensions: 92% in groundedness, and 94% for attributes that genuinely evolve over time, validating the effectiveness of our temporal conflict identification process. Further, in **Overall Correctness**—representing the proportion of questions that pass *all* three question quality dimensions simultaneously, achieves a high score of 80%. This high question quality score is complemented by 91.57% correctness rate for the curated gold answers. Taken together, these results demonstrate that evolveQA is a reliable benchmark.

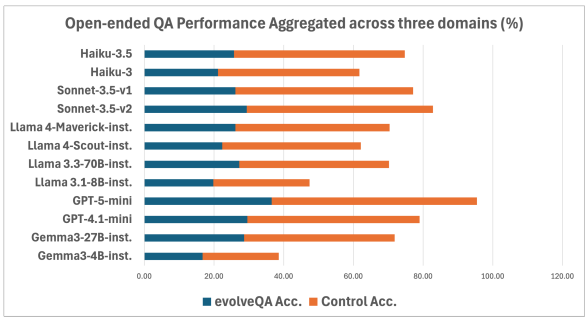

Figure 3: Performance comparison between evolveQA and control set on open-ended QA, aggregated across the three domains. We observe a significant performance degredation on evolveQA across all models.

B.6.2 Meta Evaluation of Model Judge

In a supplementary analysis, on this sample of 100 question-answer pairs, we also evaluated the performance of our model judge (`Claude Sonnet 4`). The judge achieved an accuracy of 95.9% in evaluating LLM responses against the gold answers, confirming its reliability for our experiments. We also found that in 2.7% of the cases, the judge failed to correctly identify whether the model's response was "outdated" or "incorrect". In the remaining 1.35% of the cases, the judge was completely incorrect in its decisions.

Metric	Score (%)
Question Evaluation	
Grounded in E-C-A	92
Knowledge Evolution	94
Quality & Specificity	89
Overall Correctness	80
Gold Answer Correctness	91.57

Table 6: Human evaluation results validating the quality of the evolveQA benchmark. We observe high quality on individual dimensions, such as *grounding* (92%) and targeting genuine *knowledge evolution* (94%). **Overall Correctness** is our strict composite metric, representing the proportion of QA pairs that successfully pass *all* preceding quality checks. The high score on this strict measure (80%), coupled with high gold answer correctness (91.57%), confirms the overall reliability of our framework.

B.7 Main Results: Additional Discussion

Figure 3 illustrates the performance of evolveQA when compared against the control set. For the purposes of this visualization, we have averaged model performance across all the domains. We observe that each model significantly lags behind the control set when probed for temporally conflicting information in an open-ended QA setting.

B.7.1 Notable Anomalies: `Claude-3.5-Haiku`

Here, we provide examples of our qualitative study. Figure 4 illustrates two cases from *Azure* domain, where the model (`Claude-3.5-Haiku`) generated correct response in open-ended QA and generated incorrect response in its corresponding MCQ format.

C Ablation Study Results

In this section, we provide the detailed results for the following ablation studies: (1) Outdated Response Rate (Section C.1); (2) Performance on *superseding* knowledge (Section C.2); (3) Performance with explicit context (Section C.3). In addition, to mitigate self-preference bias, we also evaluate responses from Claude models with a different model-judge– `GPT-4.1-mini` (Section C.4).

C.1 Outdated Response Rate

As discussed in the ablation study in the main paper (Section 4.2), simply measuring accuracy on the open-ended QA task can mask the specific failure modes of LLMs on `evolveQA`. To provide a more granular analysis, we categorize each generated response into one of three categories: (a) **Accurate:** The response is factually correct and aligns with the most current gold-standard answer. (b) **Outdated:** The response is factually correct for a past time period but is now outdated or has been superseded. This is the Outdated Response Rate (`ORR`). (c) **Incorrect:** The response is neither accurate nor outdated. Table 7 presents the full breakdown for all models on the open-ended QA task. The key observation is the consistently high `ORR`, which confirms that a primary failure mode is the retrieval of stale, parametric knowledge. We observe that for certain models, the proportion of outdated responses reaches as high as 55% in the *AWS* domain, 50% in the *Azure* domain, and 39% in the *WHODONs* domain. This data provides strong evidence that LLMs frequently mistake deeply ingrained yet obsolete information for current fact.

C.2 Superseding Knowledge

Table 8 provides an example of *additive* vs. *superseding* knowledge. In the case of *additive* knowledge, there could potentially be multiple correct answers. As a result, model's have a higher chance of getting them correct. However, *superseding* knowledge becomes highly challenging for the models, especially in MCQ format, where we present a previously valid response as part of the input question.

Consequently, as illustrated in Table 9, we observe a significant reduction in performance in the case of *superseding* knowledge, when compared against the entire `evolveQA` benchmark.

C.3 Providing Explicit *Entity, Concept, Attribute* Context

Table 10 shows the results from the cases where we provide explicit contextual$-\{$*entity, concept, attribute*$\}$ and temporal hints (through a "current date" $-$ which is a randomly chosen date before the model's knowledge cut-off date and based on the question's time-frame).

C.4 Mitigating Self-Preference Bias: Evaluating Claude Responses with `GPT-4.1-mini`

To evaluate open-ended responses, our methodology uses `Claude Sonnet 4` (from *Anthropic family*) as the model judge. We utilize this model-judge even to evaluate responses from other *Anthropic family* of models such as `Claude 3.5 Sonnet, Haiku`. Therefore, to empirically validate if this model-judge has a tendency to prefer responses from its own family, we evaluate open-ended responses from *Anthropic family* of models using an additional model-judge$-$`GPT-4.1-mini`. While the absolute numbers varies due to the different judge, even with this judge, we observe a consistent pattern (illustrated in Table 11), as discussed in the main paper.

D Prompt templates

In this section, we provide all the prompt templates utilized for the purposes of benchmark creation and evaluation.

D.1 Prompts Related to Benchmark Creation

In this section, we provide the prompt templates used for creating our benchmark, `evolveQA`. Specifically, Figure 5 provides the prompt template for *concept and salient entity extraction*; Figure 6 shows the prompt template for *coherency check*; Figure 7 for *cluster label generation*; Figure 8 for *redundancy check*; Figures 10, 11 shows the prompt templates for *question generation*; Figures 9, 12 shows the prompt templates for *answer curation*.

EXAMPLE - 1

Open-ended QA:
Question: What storage tier is supported for creating incremental snapshots of Ultra Disk in the preview release?
Gold Answer: Standard HDD

Claude-3.5-Haiku Generated Answer:
In the preview release, Standard HDD storage tier is supported for creating incremental snapshots of Ultra Disk in Azure. This allows users to efficiently capture point-in-time backups of Ultra Disk volumes while utilizing the more cost-effective Standard HDD storage for snapshot storage.

Multiple-Choice Question:
Question: Which Azure storage tier supports incremental snapshots of Ultra Disk during the preview phase?
Options: "A": "Standard SSD" "B": "Ultra SSD" "C": "Premium SSD" "D": "Standard HDD"
Gold Answer Option: "A"
Claude-3.5-Haiku Generated Answer: "B"

EXAMPLE - 2

Open-ended QA:
Question: What is the maximum Fault Domain count allowed when deploying specialty VM types against a Capacity Reservation?
Gold Answer: "1"

Claude-3.5-Haiku Generated Answer:
When deploying specialty VM types against a Capacity Reservation in Azure, the maximum Fault Domain count is limited to 1. This means that for specialty VM types, the Capacity Reservation will only support deployments across a single Fault Domain, which restricts the high availability and fault tolerance options compared to standard VM deployments.

Multiple-Choice Question:
Question: What is the maximum Fault Domain count allowed when deploying specialty VM types against a Capacity Reservation?
Options: "A": "4" "B": "1" "C": "3" "D": "2"
Gold Answer Option: "B"
Claude-3.5-Haiku Generated Answer: "A"

Figure 4: Shows two examples where `Claude-3.5-Haiku` generates correct response in open-ended QA whereas, it generates an incorrect response for its corresponding MCQ format in the control set.

D.2 Prompts Related to Different Knowledge Probing Formats

Figure 13, 14 shows the prompts used for obtaining corresponding MCQ and Verifiable QA for an open-ended question-answer pair in `evolveQA`.

D.3 Evaluation Rubric (Prompt) for Model Judge

Figure 15 shows the prompt template used by the model judge to evaluate responses from all the LLMs on open-ended questions in `evolveQA`.

Domain	Model Family	Model	evolveQA (%)			Control Set (%)
			Acc.	Outdated (ORR)	Incorrect	Acc.
AWS Whatsnew Feed	Anthropic	Claude 3.5-sonnet-v2	26.95	50.21	22.84	46.39
		Claude 3.5-sonnet-v1	22.9	51.33	25.77	44.41
		Claude 3 Haiku	15.93	52.46	31.62	33.84
		Claude 3.5 Haiku	22.64	54.91	22.45	42.14
	Llama	Llama 3.1-8B-Instruct	14.44	42.89	42.67	20.33
		Llama 3.3-70B-Instruct	18.53	47.63	33.84	33.13
		Llama 4-Scout-instruct	17.09	51.45	31.45	31.78
		Llama 4-Maverick-instruct	22.18	47.64	30.18	36.45
	OpenAI	GPT-4.1-mini	22.86	54.67	22.47	41.69
		GPT-5-mini	29.62	42.94	27.44	55.4
	Gemma	Gemma3-4B-instruct	12.18	36.73	51.09	18.83
		Gemma3-27B-instruct	21.82	46.18	32.0	34.71
Azure Updates	Anthropic	Claude 3.5-sonnet-v2	22.95	50.27	26.78	49.82
		Claude 3.5-sonnet-v1	21.31	46.99	31.69	46.24
		Claude 3 Haiku	18.18	46.75	35.06	27.1
		Claude 3.5 Haiku	21.32	48.22	30.46	41.5
	Llama	Llama 3.1-8B-Instruct	17.58	34.55	47.88	16.85
		Llama 3.3-70B-Instruct	24.24	44.24	31.52	36.26
		Llama 4-Scout-instruct	18.72	48.82	33.0	31.58
		Llama 4-Maverick-instruct	23.15	46.31	30.54	37.19
	OpenAI	GPT-4.1-mini	26.84	40.53	32.63	38.35
		GPT-5-mini	29.47	45.26	25.26	49.82
	Gemma	Gemma3-4B-instruct	14.29	30.54	55.17	15.09
		Gemma3-27B-instruct	25.12	39.41	35.47	30.88
WHO-Disease Outbreak News	Anthropic	Claude 3.5-sonnet-v2	38.41	39.07	22.52	64.09
		Claude 3.5-sonnet-v1	34.23	38.93	26.85	62.43
		Claude 3 Haiku	29.41	38.97	31.62	60.69
		Claude 3.5 Haiku	33.33	37.74	28.93	63.44
	Llama	Llama 3.1-8B-Instruct	27.4	38.36	34.25	45.71
		Llama 3.3-70B-Instruct	39.04	37.67	23.29	59.43
		Llama 4-Scout-instruct	31.36	44.97	23.67	55.91
		Llama 4-Maverick-instruct	33.14	37.87	28.99	59.14
	OpenAI	GPT-4.1-mini	39.1	35.26	25.64	68.13
		GPT-5-mini	50.64	30.77	18.59	71.43
	Gemma	Gemma3-4B-instruct	23.67	23.08	53.25	31.72
		Gemma3-27B-instruct	39.05	31.95	28.99	63.98

Table 7: Breakdown of performance on evolveQA Open-Ended QA. Here, we report: Accuracy, *Outdated Response Rate* (ORR), and Incorrect response (%), across datasets. Performance on non-conflicting control set is also reported. We observe a high ORR, at least 23%, across all datasets.

Additive Knowledge		Superseding Knowledge	
Question	**Answers (all valid)**	**Question**	**Answer**
How does Amazon Kendra obtain user and group access control information for search filtering?	through secure search tokens... through its principal store... AWS SSO identity store...	How many AWS regions globally support Amazon Kendra?	4 5 7

Table 8: This table provides examples of questions that test for two distinct types of knowledge: *Additive*−multiple valid answers, and *Superseding*− only most recent answer is valid.

Domain	Model Family	Model	MCQ: Acc. (%)			Open-Ended QA: Acc. (%)		
			`evolveQA - superseding knowledge`	`evolveQA`	Control Set	`evolveQA - superseding knowledge`	`evolveQA`	Control Set
AWS Whatsnew Feed	Anthropic	Claude 3.5-sonnet-v2	51.32	63.33	82.49	25.28	26.95	46.39
		Claude 3.5-sonnet-v1	49.81	63.33	83.47	19.78	22.9	44.41
		Claude 3 Haiku	49.79	62.47	68.07	15.88	15.93	33.84
		Claude 3.5 Haiku	60.93	67.69	73.24	21.51	22.64	42.14
	Llama	Llama 3.1-8B-Instruct	53.7	61.84	67.53	15.56	14.44	20.33
		Llama 3.3-70B-Inst.	56.42	64.99	73.78	15.95	18.53	33.13
		Llama 4-Scout-inst.	58.54	68.32	75.35	16.72	17.09	31.78
		Llama 4-Maverick-inst.	53.66	66.3	79.51	21.95	22.18	36.45
	OpenAI	GPT-4.1-mini	47.76	62.78	78.24	17.54	22.86	41.69
		GPT-5-mini	47.01	59.86	82.08	20.52	29.62	55.4
	Gemma	Gemma3-4B-inst.	42.16	52.85	58.24	11.85	12.12	18.83
		Gemma3-27B-inst.	52.26	64.46	71.65	19.86	21.82	34.71
Azure Updates	Anthropic	Claude 3.5-sonnet-v2	64.44	70.33	77.73	23.33	22.95	49.82
		Claude 3.5-sonnet-v1	63.33	70.88	78.91	18.89	21.31	46.24
		Claude 3 Haiku	66.27	70.78	64.44	13.25	18.18	27.1
		Claude 3.5 Haiku	56.84	68.72	64.75	15.79	21.32	41.5
	Llama	Llama 3.1-8B-Inst.	56.18	63.64	60.4	19.1	17.58	16.85
		Llama 3.3-70B-Inst.	59.55	70.3	73.2	21.35	24.24	36.26
		Llama4-Scout-inst.	61.86	72.64	71.76	19.59	18.72	31.58
		Llama4-Maverick-inst.	68.04	76.12	75.95	24.74	23.15	37.19
	OpenAI	GPT-4.1-mini	59.14	71.81	76.56	22.58	26.84	38.35
		GPT-5-mini	52.69	67.55	76.95	26.88	29.47	49.82
	Gemma	Gemma3-4B-inst.	50.52	54.73	51.53	14.43	14.29	15.09
		Gemma3-27B-inst.	62.89	69.15	64.12	22.68	25.12	30.88
WHO–Disease Outbreak News	Anthropic	Claude 3.5-sonnet-v2	64.71	73.1	86.19	31.03	38.41	64.09
		Claude 3.5-sonnet-v1	62.35	71.72	90.61	25.88	34.23	62.43
		Claude 3 Haiku	62.03	66.67	81.5	24.05	29.41	60.69
		Claude 3.5 Haiku	70.1	72.08	80.11	28.87	33.33	63.44
	Llama	Llama 3.1-8B-Inst.	62.79	67.36	73.14	23.26	27.4	45.71
		Llama 3.3-70B-Inst.	61.63	68.75	83.43	32.56	39.04	59.43
		Llama 4-Scout-inst.	62.11	68.71	82.26	29.47	31.36	55.91
		Llama 4-Maverick-inst.	70.53	76.07	84.95	28.42	33.14	59.14
	OpenAI	GPT-4.1-mini	57.3	72.19	81.87	33.71	39.1	68.13
		GPT-5-mini	59.55	72.85	86.26	42.7	50.64	71.43
	Gemma	Gemma3-4B-inst.	56.84	63.19	64.52	18.95	23.67	31.72
		Gemma3-27B-inst.	67.37	71.17	79.51	34.74	39.05	63.98

Table 9: Performance comparison on the `superseding` knowledge subset of `evolveQA`, the full `evolveQA` benchmark, and the non-conflicting control dataset. The sharp decline in MCQ accuracy on the superseding subset highlights that historically correct but outdated answers significantly impairs the models' ability to identify current information.

Domain	Model Family	Model	Accuracy (%)			
			Open-ended QA	Open-Ended QA with Context	Open-Ended QA with Context and Date	Open-Ended QA with Date
AWS Whatsnew Feed	**Anthropic**	Claude 3.5-sonnet-v2	26.95	31.69	32.3	28.4
		Claude 3.5-sonnet-v1	22.9	27.37	28.83	26.34
		Claude 3 Haiku	15.93	21.78	24.12	16.16
		Claude 3.5 Haiku	22.64	25.66	28.49	26.23
	Llama	Llama 3.1-8B-Instruct	14.44	20.47	20.52	15.55
		Llama 3.3-70B-Instruct	18.53	23.28	26.08	20.26
		Llama 4-Scout-instruct	17.09	23.82	24.36	19.27
		Llama 4-Maverick-instruct	22.18	24.91	26.36	22.18
	OpenAI	GPT-4.1-mini	22.86	25.84	28.83	24.25
		GPT-5-mini	29.62	34	34.19	29.82
	Gemma	Gemma3-4B-instruct	12.18	18.55	20.91	14.36
		Gemma3-27B-instruct	21.82	27.09	27.09	23.82
Azure Updates	**Anthropic**	Claude 3.5-sonnet-v2	22.95	31.15	31.15	22.4
		Claude 3.5-sonnet-v1	21.31	32.24	30.05	22.95
		Claude 3 Haiku	18.18	27.27	25.97	21.43
		Claude 3.5 Haiku	21.32	30.96	30.96	23.35
	Llama	Llama 3.1-8B-Instruct	17.58	24.24	26.67	16.36
		Llama 3.3-70B-Instruct	24.24	28.48	29.7	25.45
		Llama 4-Scout-instruct	18.72	27.59	27.59	19.7
		Llama 4-Maverick-instruct	23.15	32.51	31.53	23.65
	OpenAI	GPT-4.1-mini	26.84	31.58	31.58	29.47
		GPT-5-mini	29.47	36.32	36.84	28.95
	Gemma	Gemma3-4B-instruct	14.29	21.18	20.2	15.27
		Gemma3-27B-instruct	25.12	29.06	27.09	25.12
WHO-Disease Outbreak News	**Anthropic**	Claude 3.5-sonnet-v2	38.41	39.6	46.31	39.6
		Claude 3.5-sonnet-v1	34.23	39.6	38.93	36.91
		Claude 3 Haiku	29.41	30.15	33.82	31.62
		Claude 3.5 Haiku	33.33	34.59	32.7	36.48
	Llama	Llama 3.1-8B-Instruct	27.4	28.08	26.03	27.4
		Llama 3.3-70B-Instruct	39.04	36.99	39.04	34.93
		Llama 4-Scout-instruct	31.36	29.59	30.77	29.59
		Llama 4-Maverick-instruct	33.14	33.14	33.14	31.95
	OpenAI	GPT-4.1-mini	39.1	39.74	41.03	37.18
		GPT-5-mini	50.64	49.36	53.21	46.15
	Gemma	Gemma3-4B-instruct	23.67	26.04	22.49	28.4
		Gemma3-27B-instruct	39.05	34.32	37.28	40.24

Table 10: Ablation study on the impact of providing contextual and temporal hints to LLMs in the open-ended QA setting of `evolveQA`. The table presents the accuracy of each model across the three domains under four conditions, comparing the baseline (no context) to performance when provided with context (the {*entity, concept, attribute*} tuple), a date hint, or both. The date hint is sampled relative to the model's knowledge cut-off date.

Domain	Model Family	Model	Non-Conflicting Control Dataset [Open-Ended QA (%)]		evolveQA [Open-Ended QA (%)]					
			Sonnet-4 Judge	GPT-4.1-mini Judge	Sonnet-4 Judge			GPT-4.1-mini Judge		
			Acc.	Acc.	Acc.	Outdated	Incorrect	Acc.	Outdated	Incorrect
Azure	Anthropic	Claude 3.5-sonnet-v2	49.82	54.12	22.95	50.27	26.78	28.89	47.78	23.33
		Claude 3.5-sonnet-v1	46.24	51.61	21.31	46.99	31.69	20.00	56.67	23.33
		Claude 3 Haiku	27.10	33.21	18.18	46.75	35.06	25.30	53.01	21.69
		Claude 3.5 Haiku	41.50	43.66	21.32	48.22	30.46	25.26	51.58	23.16
AWS	Anthropic	Claude 3.5-sonnet-v2	46.39	50.92	26.95	50.21	22.84	29.43	59.62	10.94
		Claude 3.5-sonnet-v1	44.41	49.65	22.90	51.33	25.77	25.94	60.90	13.16
		Claude 3 Haiku	33.84	37.69	15.93	52.46	31.62	21.94	62.87	15.19
		Claude 3.5 Haiku	42.14	46.88	22.64	54.91	22.45	26.79	59.29	13.93
WHO	Anthropic	Claude 3.5-sonnet-v2	64.09	69.61	38.41	39.07	22.52	40.00	41.18	18.82
		Claude 3.5-sonnet-v1	62.43	64.64	34.23	38.93	26.85	42.35	40.00	17.65
		Claude 3 Haiku	60.69	56.65	29.41	38.97	31.62	29.11	41.77	29.11
		Claude 3.5 Haiku	63.44	63.44	33.33	37.74	28.93	37.11	21.65	41.24

Table 11: Shows accuracy scores for the Claude-family models on open-ended `evolveQA`, evaluated by our primary judge (`Claude Sonnet 4`) and cross-validated with an out-of-family judge (`GPT-4.1-mini`). The consistent trends across both judges demonstrate the robustness of our evaluation methodology.

Prompt: Shows the prompt used for *concept* and *salient entity* extraction.

```
### Task:
You are a meticulous research analyst. Your task is to identify the **key concepts or salient topics** introduced or emphasized in the
      provided document.

### Instructions:
- First, thoroughly analyze the document to identify its most significant topics. A topic is considered **salient** if it represents the
      document's **central thesis or primary purpose.** It should be the main reason the document exists, such as a core argument, a
      significant finding, a key announcement, or a central event being described. **Disregard background information, and general
      statements.**
- Identify **up to {k}** such salient topics. It is crucial that you do not invent topics simply to meet this quota. If a document contains
      only one essential point, identify only that one. If there are no salient topics, you must identify none.
- For each salient topic you identify (if any), you are required to provide the following:
   - **Normalized Concept Title:** This title should encapsulate the main idea of the topic by omitting specific details such as names of
         people, organizations, products, or specific dates and numbers. The goal is to generalize the concept to its fundamental theme,
         capturing the key essence of the information presented. It should **highlight what is being introduced or emphasized**, not where
         or when.
   - **Salient Entity:** This is the primary subject associated with the identified topic. Provide its **canonical name** to ensure it is
         generalizable and consistently identifiable across different documents (e.g., use "General Data Protection Regulation" instead of
         "GDPR", use "United Nations" instead of "UN Office of the High Commissioner for Refugees").

### Strict Requirements:
- **Strictly ensure that the identified concepts are well-grounded in the provided document.**
- **Ensure that each concept is distinct and not a repetition of another.**
- **Ensure each concept is salient and not a minor detail or background information.**
- **Ensure that each entity associated with a concept is a salient entity, not a minor or peripheral mention.**
- **Ensure that each entity is a canonical name, not an abbreviation or informal reference.**

### Strict JSON Output Format:
```json
{{
"1": {{
   "normalized_title": "insert-normalized-title-here",
   "salient_entity": "insert-entity-name-here"
   }}
}}
```
**If no salient topics are found, the value must be an empty json dict `{{}}`. Do not include any other text or explanation outside of the
      JSON list.**

Example Input Document:
The Environmental Protection Agency today issued a new regulation, 'Clean Water Mandate 2025,' which sets stricter limits on the industrial
      discharge of phosphates into the nation's rivers. This rule, which takes effect in January, aims to combat the recent surge in harmful
      algal blooms that have plagued major waterways. The administrator of the agency, Jane Smith, stated that the 'Clean Water Mandate
      2025' is a critical step in preserving aquatic ecosystems for future generations.

Example Output:
```json
{{
"1": {{
       "normalized_title": "Environmental agency issues new water quality regulation",
       "salient_entity": "Environmental Protection Agency"
   }},
"2": {{
       "normalized_title": "Implementation of stricter limits on industrial phosphate discharge",
       "salient_entity": "Clean Water Mandate 2025"
   }}
}}
```

Input Document:
{input_document}
```

Prompt: Shows the prompt used for *coherency check*.

```
### Task:
You are an expert at identifying thematic connections. Your
      task is to analyze the provided input documents, and
      determine whether **all the documents** are discussing
      the same **high-level concept**.

**Instructions**:
- Note that **all the documents are from the same domain**,
      but the focus is on whether they share a *precise
      high-level concept*.
- For example, if one document discusses *`new feature for
      data analysis is released`* and another *`price
      reduction for a service`*, they cover different
      *high-level concepts*.

**If all the documents are about the same `high-level
      concept`, then respond with a `"yes"`.**

### Strict JSON Output Format:
```json
{{"same": <"yes"/"no">}}
```
Strictly do not output any other information.

Input Documents:
{input_docs}
```
```

Prompt: Shows the prompt used for *concept-cluster label generation*.

```
### Task:
Analyze the provided input documents, and come up with a
      concise `title` that accurately reflects the content of
      these documents.

### Strict JSON Output Format:
```json
{{"title": "insert-title-here"}}
```
Strictly do not output any other information.

Input Documents:
{input_docs}
```
```

Prompt: Shows the prompt used for *redundancy check*.

```
### Task:
You are given two `labels`. Your task is to determine if these
      `labels` are discussing the **same** concepts and
      should be merged.

### Strict JSON Output Format:
```json
{{"merge": <"yes"/ "no">}}
```
Strictly do not output anything other information.

Input:
`Label 1`: {label1}
`Label 2`: {label2}
```
```

Prompt: Prompt used to check evidence sufficiency (first step of answer generation).

```
### Task:
You are given a `document` and a `question`. Determine if the
      `document` is **relevant** to answering the `question`.

**Instructions:**
1. Read the `document` carefully.
2. Analyze the `question` to understand what information is
      being asked.
3. Check if the `document` is **directly related** to the
      `question`. It **must talk about the same topic** or
      provide information that can help answer the question.
4. If the `document` contains **relevant** information to the
      `question`, respond with **yes**.

### Strict JSON Output Format:
```json
{{
"relevant": <"yes"/"no">
}}
```
Strictly do not output any other information.

Input:
`**Document**`: {document_content}

`**Question**`: {question}
```
```

Prompt: Shows the prompt used for *identifying evolving knowledge*.

```
### Task:
You specialize in detecting *how* knowledge changes over time. Your specific task is to determine if a set of documents exhibits **knowledge
      evolution**.

**Knowledge evolution** refers to a meaningful change in the state of knowledge over time, where later documents **update, revise, correct,
      or meaningfully alter** earlier information, such that previous understanding becomes outdated, incomplete, or less accurate. This is
      distinct from a purely **additive change**, where new information is introduced without invalidating earlier knowledge.

You will be provided:
- An `entity`: the concrete item under discussion (e.g., Amazon EC2).
- A `concept`: the abstract topic this entity belongs to (e.g., service regional availability).
- An `attribute`: the specific discussion aspect being tracked (e.g., cost associated with the service).
- A chronological list of **representative** `documents`, each tagged with a `timestamp`.

Your task is to analyze the documents and determine if there is **any evidence of knowledge evolution** with respect to the **combination of
      `entity`, `concept`, and `attribute`.** Strictly focus on the intersection of these three components.

### What typically constitutes knowledge evolution:
Knowledge evolution typically involves:
- A specific **fact or value is updated** over time (e.g., revised statistics, corrected measurements).
- A recommended **method, policy, or process changes** (e.g., an older guideline is replaced by a newer one).
- A **renaming, redefinition, or reclassification** of an entity or attribute.
- An earlier statement is **explicitly corrected, contradicted, or substantively refined** by a later document.

### What is NOT knowledge evolution:
We provide typical examples of what does **not** constitute knowledge evolution below:
- **Purely additive changes.** For example, if a **previously unavailable feature becomes available**. This represents a new addition,
      **not** a knowledge evolution that invalidates earlier information.
- **Stable knowledge**, where later documents do not conflict with, revise, or meaningfully change earlier statements.

### Strict Requirement:
**If knowledge evolution is observed, return "yes". Otherwise, return "no".**

### Strict JSON Output Format:
**If you determine knowledge evolution exists, return:**
```json
{{
  "evolving_knowledge": "yes",
  "reasoning": "A concise explanation of how the knowledge evolved over time. Clearly state what the earlier understanding was and how later
        documents updated, corrected, or revised it. The reasoning must be strictly grounded in the provided documents."
}}
```
If you cannot determine any superseding fact, return:
```json
{{
  "evolving_knowledge": "no",
  "reasoning": "No knowledge evolution was found. The knowledge is either stable or the evolution is purely additive."
}}
```

Strictly do not output any other information.

Input:
`Entity`: {ent_name}
`Concept`: {concept_name}
`Attribute`: {attribute_name}
`Representative Documents` with timestamp:\n{document_chunks_with_timestamps}
```

Prompt: *Question Generation* prompt template.

```
### Follow-Up Task:
Evolving knowledge has been detected. Your task is to generate a single, **well-grounded** question that probes a **specific point of
      knowledge whose value or state has been updated or superseded over time.**

### Core Principle:
Your goal is to frame a question where **multiple documents could provide an answer, but their answers conflict or are outdated.** The most
      recent document must provide the definitive, correct answer that supersedes all previous ones.

### Strict Requirements:
- The question must be **strictly grounded in the provided `entity`, `concept`, and `attribute`**.
- It must **strictly target a single, specific point of knowledge.**
- It must be **precise and unambiguous**.
    - Do **not** include temporal information such as `month`, `year`, or `time period` in the question.
    - Do **not** ask trivial questions. For instance, "What verb is used in describing ...". It is trivial and does not capture any
          meaningful change in knowledge.
- Strictly **ensure** that the question does **not** elicit a list of knowledge or features. For instance, do **not** ask "What types of
      changes have been made to the service?". Here, the user is expected to list a few changes or all changes, which is **strictly
      prohibited**. Instead, focus on a **specific change** that has occurred. Another **bad example** is "In which regions is the service
      now available?". Again, **this is a list question and strictly prohibited.**

### Framing Hints:
- **Look for replacements, not just knowledge additions.** Search for facts or values that are explicitly changed or made obsolete by newer
      documents.
- **To help you identify these, look for common patterns of superseding knowledge. Typical examples include:**
    - **A numerical value**, like a limit, size, count, or other key metric.
    - A **recommended process or standard**, where a new method replaces an old one.
    - A **formal name or title** that is officially changed or rebranded.
    - An **official status or classification**, like a rating, phase, or category.
- **The ideal question is one that would cause a user to be wrong if they only had access to the older documents.**

### Strict JSON Output Format:
```json
{{
  "question": "insert your question here",
  "reasoning": "A brief explanation of why this question probes evolving knowledge. Specifically state what the old answer would be and what
        the new, correct answer is, and why this represents a superseding change."
}}
```
If you cannot frame a question that meets the Core Principle, return empty strings::
```json
{{
  "question": "",
  "reasoning": "No question could be framed because the documents only show knowledge being added (like items to a list) rather than
        knowledge being updated or superseded."
}}
```
Strictly do not output anything else.
```

Prompt: *Answer Generation* prompt.

```
### Task:
Your goal is to synthesize a final, comprehensive answer to a given question based on multiple responses from different sources.
Each answer includes:
- An `id`
- A `timestamp`
- The `answer_text`
- Some `Additional Context from Document` (for reference)

These answers are ordered by timestamp in **descending order** (i.e., most recent first).

### Instructions:
1. **Understand the Question**: Carefully read the question to identify the key information being asked.
2. **Analyze All Answers**: Consider both the content and timestamp of each answer. Determine which answers provide up-to-date, and relevant
      information.
3. **Handle Temporal Evolution Thoughtfully**:
   - If newer answers clearly correct or replace earlier ones, prioritize the most recent valid information.
   - However, if earlier answers provide **complementary or still-relevant information** that has not been explicitly invalidated, include
         them as well.
   - Avoid discarding older responses **unless** they are clearly outdated, incorrect, or contradicted by newer ones.
4. **Synthesize the Final Answer**:
   - Write a concise and coherent response that reflects the most accurate answer to the question.
   - Use only relevant content.
   - In some cases, a single recent answer may be sufficient. In others, multiple answers may need to be combined to fully answer the
         question.
5. **Exclude outdated or irrelevant answers** if their information is no longer valid or is contradicted by newer responses.
6. **Ground Your Answer**: Ensure your final synthesized answer is **grounded in the provided source answers**. Do not add any information
      not present in the source answers.

### Strict JSON Output Format:
```json
{{
"final_answer": <"Your synthesized answer to the question">,
"answer_ids": <[list of **ALL** answer ids used in synthesis]>,
}}
```
Strictly do not output any other information.

Input:
`**Question**`: {question}

`**Answers (most recent first)**`:
{answers}
```

Prompt: Prompt used for obtaining corresponding MCQ format for an open-ended question.

```
### Task:
You are given the following inputs:
- A `question`: A natural language query that tests specific knowledge.
- A `gold_answer`: The correct answer to that question.
- A list of `distractor_chunks` (numbered 1-4): These are pieces of related but **misleading** information. You will use them to create
       **plausible but incorrect** answer choices (called distractors). These distractors must be hard-to-guess and grounded in the provided
       chunks.

Your goal is to construct a **4-option multiple-choice question (MCQ)** using these inputs.

### **Strict** Guidelines:
1. **Knowledge Equivalence**:
   - The MCQ must test the **same knowledge** as the original `question`.
   - You may rephrase the question, but you **must not** change its intent or focus.
2. **Correct Answer Usage**:
   - Use the provided `gold_answer` as the **only correct answer**.
3. **Timeline Constraint - No Explicit Dates**:
  - Do **not** include any explicit dates, months, or years (e.g., “May 2023”, “in June”, “2022”) in the generated MCQ question.
  - Strictly ensure that the question remains very **specific, concise, and unambiguous**.
4. **Clarity and Precision Requirement**:
   - The MCQ **question itself** must be:
       - **Specific** - Ask about a clearly defined piece of knowledge as the original question
       - **Concise** - Avoid unnecessary words or complexity as the original question
       - **Unambiguous** - Have **only one** reasonable interpretation
  - If the question violates this, return the fallback format instead.
5. **Distractor Construction**:
   - Construct **three incorrect answer options** (distractors) using content **only from the `distractor_chunks`**.
   - Each distractor must be:
       - Plausible (not obviously wrong),
       - Clearly incorrect,
       - **Well-Grounded** in the corresponding chunk.
6. **Chunk Mapping Requirement**:
   - For each distractor (A, B, C, or D), indicate **which `distractor_chunk` (1-4)** it was based on.
   - Use the `used_chunks_for_distractors` dictionary to record this mapping.
        - For example: `"1": "B"` means distractor B was created using distractor chunk 1.
   - The correct option (matching the gold answer) should not have a mapped chunk.
7. **Special Case - Add One Extra Numeric Distractor**:
   - If the question elicits a **numeric answer** (e.g., involving terms like `highest`, `most`, `longest`, `maximum`, `recent`, `lowest`,
         etc.):
       - You **must** generate **one additional numeric distractor** that:
           - Is **greater** than the value in the `gold_answer`
           - Retains the same **unit or measurement** (e.g., "days", "kilometers", "%")
       - Store this value in an additional field called `"additional_distractor"`.
   - This field is **optional** —it should only be included **only if** the rule applies (when the question elicits a numeric answer).
   - Do **not** insert this distractor into the main options A-D.
8. **Fallback Behavior**:
   - If you cannot generate a valid MCQ, return the fallback JSON structure with all fields empty.

### Required JSON Output Format:
If a valid MCQ **is successfully created**:
```json
{{
  "mcq_question": "Your multiple-choice question here",
  "options": {{
    "A": "Option A text", "B": "Option B text", "C": "Option C text", "D": "Option D text"
  }}, "gold_answer_option": "A", "used_chunks_for_distractors": {{"2": "C", "3": "B", "4": "D"}},
"additional_distractor": "insert numeric distractor here (if applicable)"}}
```
Strictly do not output any other information.

Fallback Output Format (if no MCQ possible):
```json
{{
  "mcq_question": "",
  "options": {{
    "A": "", "B": "", "C": "", "D": ""
  }}, "gold_answer_option": "", "used_chunks_for_distractors": {{}}, "additional_distractor": ""}}
```
Strictly do not output any other information.

Input:
`Question`: {question}
`Gold_Answer`: {answer}
`Distractor_Chunks`: {distractor_chunks}
```

Prompt: Prompt used for obtaining corresponding Verifiable QA for an open-ended question.

```
### Task:
You are given two inputs:
- A `question`: A natural language query that tests factual knowledge.
- An `answer`: The correct answer to that question.

Your goal is to create **two questions - a 'yes' question and a 'no' eliciting question**, by embedding the answer directly into the
      question (if possible). These questions should be verifiable as either **true** or **false**.

### Strict Instructions:
1. **Answer Embedding**:
  - Try to rewrite the input question by **embedding the provided answer** directly into the question text. This creates a fact-like
        question.
    - Example:
      - Example Input:
        - `question`: What city is the capital of Canada?
        - `answer`: Ottawa
      - Example Output 1:
        - `yes_question`: Is Ottawa the capital of Canada?
        - `no_question`: Is Toronto the capital of Canada?
      - Example Output 2:
        - `yes_question`: Ottawa is the capital of Canada. Is this correct?
        - `no_question`: Ottawa is not the capital of Canada. Is this correct?
  - **Generate the following two Versions**:
    - `yes_question`: A 'yes' response eliciting version.
    - `no_question`: A 'no' response eliciting version. It must be a plausible alternative that is **not true** based on the original
        question and answer.

2. **Knowledge Equivalence**:
  - Both question versions must test the **exact same knowledge** as the original `question`.
  - You may rephrase the question, but you **must strictly not** change its intent or focus.

3. **Clarity and Precision Requirement**:
  - The generated **questions themselves** must be:
    - **Specific** - Ask clearly defined piece of knowledge as the original question
    - **Concise** - Avoid unnecessary words or complexity as the original question
    - **Unambiguous** - Have **only one** reasonable interpretation
  - If the question violates this, return the fallback format instead.

4. **Timeline Constraint - No Explicit Dates**:
  - Do **not** include any explicit dates, months, or years (e.g., “May 2023”, “in June”, “2022”) in the generated questions.
  - Strictly ensure that the question remains very **specific, concise, and unambiguous**.
  - Example:
    - Example Input:
      - `question`: What is the primary storage mechanism for Azure Blob backups in Azure Backup?
      - `answer`: As of the most recent update (March 2023), the primary storage mechanism for Azure Blob backups in Azure Backup is a
          vault, which is a logical entity that stores the backups and recovery points created over time. This represents a change from
          the earlier approach (prior to 2023) where backup data was stored locally within the source storage account itself.
    - **Valid Output:**
      - `yes_question`: Is a vault the primary storage mechanism for Azure Blob backups in Azure Backup?
      - Note that the `yes_question` does not include any explicit dates or months. It tests the **same knowledge** as the original question
          without introducing any new facts or assumptions. It is also **specific, concise, and unambiguous**.
    - **Invalid Output:**
      - `yes_question`: Is a vault the primary storage mechanism for Azure Blob backups in Azure Backup as of March 2023?

5. **Fallback Behavior**:
  - If you cannot rephrase the original question to include the answer naturally, strictly leave both `yes_question` and `no_question` as
        empty strings.
  - If the generated questions violate any of the above rules, strictly leave both `yes_question` and `no_question` as empty strings.

### Strict JSON Response Format:
**If a valid verifiable question pair is successfully created**, strictly return the following JSON:
```json
{{
  "yes_question": "Embedded yes-question here",
  "no_question": "Embedded no-question here",
  "evidence": "Explain how these questions are grounded in the original question and answer. Provide reasoning for knowledge equivalence,
        timeline constraints, and ambiguity requirements."
}}
```
If no valid verifiable questions can be created, return:
```json
{{
  "yes_question": "",
  "no_question": "",
  "evidence": "Explain why no valid verifiable questions could be created, such as inability to embed the answer naturally or violation of
        clarity/precision requirements."
}}
```
Strictly do not output any other information.

Input:
`Question`: {question}
`Answer`: {answer}
```

Prompt: Shows the evaluation rubric used by `Claude Sonnet 4` (judge) to evaluate responses open-ended responses from LLMs on `evolveQA`.

```
### Task:
You are an expert judge. Your task is to evaluate a model's `Generated Answer` against a definitive `Gold Answer`. You must determine if the
       `Generated Answer` correctly states the single, most authoritative, time-sensitive fact, as defined by the provided context and the
       `gold answer`.

### Critical Context and Role of Inputs:
- **`Gold Answer`**: This is the **absolute ground truth**. It contains the most recent and authoritative fact. Your primary goal is to
       check if the generated answer aligns with the core evolving fact in this answer.
- **`Question`**: This tells you what is the exact knowledge that should be elicited from the model. It is intended to elicit a specific,
       time-evolving fact.
- **`Evolving Answers`**: This is your additional **focusing tool**. Analyze this list along with the **`Question`** to pinpoint *what
       specific fact* is changing over time. This coupled with the `Question` tells you what knowledge to look for in the `Generated Answer`
       and the `Gold Answer`.
- **`Generated Answer`**: The text to be judged for accuracy against the `Gold Answer`.

### Verdict Rubric:
- **CORRECT_AND_CURRENT**: The generated answer correctly states the most recent, authoritative fact as found in the `Gold Answer`.
- **OUTDATED**: The generated answer states a fact that is not the current one, but *does* match an older fact from the `Evolving Answers`
       list.
- **INCORRECT**: The generated answer states a fact that is neither the current authoritative fact nor a previously correct one.

### Step-by-Step Evaluation Process:
You **must** follow these steps precisely to form your judgment.

1. **Prior Context**: Analyze the given **entity**, **concept**, and **attribute** to obtain the **basic** context associated with the
       `Question`.
2. **Understand the `Question`**: Analyze the `Question` to determine what specific fact it is asking about. This fact typically relates to
       an evolving piece of information.
3. **Identify and isolate the evolving fact from the `gold answer` and `generated answer`:** Analyze the provided `Evolving Answers` (along
       with the **timestamp**) to determine the **exact evolving fact** that the `Question` is asking for. This will help you to understand
       what exactly to look for in the `Gold Answer` and the `Generated Answer` (with respect to the `Question`).
4. **Classify and Justify:** Based on the detailed analysis from the previous steps, classify the `Generated Answer` into one of the three
       verdicts:
    - Compare the isolated evolving fact (from step 3) from `Generated Answer` to the isolated evolving fact in the `Gold Answer` that is
          relevant to the `Question`. If they match, the verdict is **CORRECT_AND_CURRENT**.
    - If they do not match, check if the isolated fact from the `Generated Answer` matches any of the older values in the `Evolving Answers`.
          If it does, the verdict is **OUTDATED**.
    - If it matches neither, or if the fact was missing, the verdict is **INCORRECT**.
5. **Formulate Rationale:** Your rationale must clearly **explain** your reasoning for the verdict.

### Strict JSON Response Format:
```json
{{
    "verdict": "CORRECT_AND_CURRENT | OUTDATED | INCORRECT",
    "rationale": {{
        "isolated evolving fact from gold answer": "The specific information that is changing.",
        "classification_reasoning": "A brief explanation for the verdict.",
    }}
}}
```
Strictly do not output any other information beyond the JSON.

Input:
`Question`: {question}
`Gold Answer`: {gold_answer}
`Generated Answer`: {generated_answer}
`Entity`: {entity}
`Concept`: {concept}
`Attribute`: {attribute}
`Evolving Answers`: {evolving_answers}
```

Prompt: Prompt used for validating if the answer really evolves over time.

```
### Task:
You specialize in detecting *how* knowledge changes over time. Your specific task is to determine if a set of `answers` to a particular
      `question` exhibits **knowledge evolution**.

**Knowledge evolution** refers to a meaningful change in the state of knowledge over time, where later information **updates, revises,
      corrects, or meaningfully alters** what was previously stated, such that earlier information becomes outdated, incomplete, or less
      accurate.

You will be provided:
- A `question`
- A chronological list of `answers` for that `question`, each associated with a `timestamp`

Your task is to analyze the `answers` and determine if there is **any evidence of knowledge evolution** with respect to the `question`.

**Strictly focus on the provided information —`question`, `answers`, and their `timestamps`.**

### What typically constitutes knowledge evolution:
Knowledge evolution typically involves a substantive update or shift in understanding across answers. Examples include:
- A specific **fact or value is updated** over time (e.g., revised statistics, corrected measurements).
- A recommended **method, guideline, or practice changes** (e.g., an older recommendation is replaced by a newer one).
- A **reframing, renaming, or reclassification** of an entity (e.g., a system, policy, or concept is renamed or redefined).
- An earlier statement is **explicitly corrected, contradicted, or refined** by later information.

### What is NOT knowledge evolution:
We provide a typical example of what does **not** constitute knowledge evolution below:
- **Stable or consistent knowledge over time**, where later answers do not conflict with, revise, or meaningfully extend earlier ones.

### Strict Requirement:
**If knowledge evolution is observed, return "yes". Otherwise, return "no".**

### Strict JSON Output Format:
**If you determine knowledge evolution exists, return:**
```json
{{
  "evolving_knowledge": "yes",
  "reasoning": "A concise explanation of how the knowledge evolved over time. Clearly state what the earlier understanding was and how later
        answers updated, corrected, or revised it. The reasoning must be strictly grounded in the provided answers and timestamps."
}}
```
If you cannot determine any superseding fact, return:
```json
{{
  "evolving_knowledge": "no",
  "reasoning": "No knowledge evolution was found."
}}
```
Strictly do not output any other information.

Input:
`Question`: {question_text}
A List of `Answers` with timestamp:\n{answers_with_timestamps}
```