ExAnte: A Benchmark for Ex-Ante Inference in Large Language Models

Anonymous ACL submission

Abstract

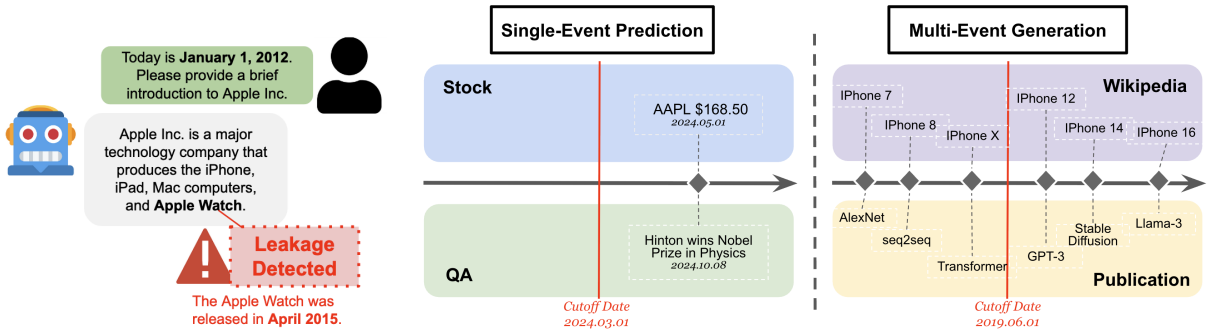
Large language models (LLMs) struggle with ex-ante reasoning—making inferences or predictions without access to future information. Even under explicit temporal cutoffs, they often rely on internalized post-cutoff knowledge. To systematically evaluate this issue, we introduce a benchmark that assesses LLMs' ex-ante inference ability across four tasks: stock prediction, question answering, Wikipedia event generation, and scientific publication generation. We quantify temporal leakage using a leakage rate metric, which measures models' reliance on future information beyond cutoff timestamps, and a quality measure that evaluates task performance. Experimental results show that LLMs frequently violate temporal constraints across tasks, revealing persistent challenges in ex-ante reasoning. Our benchmark serves as a rigorous testbed for studying temporal reasoning in time-sensitive contexts and provides complete datasets, results, and evaluation resources to support future research on improving temporal consistency in modern LLMs.

1 Introduction

Large language models (LLMs) have significantly advanced core tasks in natural language processing (NLP), demonstrating strong performance in question answering (Mallen et al., 2022; Zhuang et al., 2023), summarization (Goyal et al., 2022; Zhang et al., 2024), and reasoning (Wei et al., 2022; Lightman et al., 2023). However, ensuring that LLMs can reason under *strict temporal constraints* remains an open challenge (Yuan et al., 2024a; Fatemi et al., 2024). In many real-world applications, models must answer time-sensitive queries using only information available up to a specified cutoff date, without incorporating knowledge from events that occurred afterward. We refer to this ability as ***ex-ante inference***, a fundamental yet underexplored temporal reasoning task. This notion parallels research in human cognition on mitigating *hindsight bias*—the tendency to perceive past events as more predictable after outcomes are known (Fischhoff, 1975; Hawkins and Hastie, 1990)—which similarly examines how individuals form beliefs before outcomes are revealed (Fisher and Geiselman, 1992).

LLMs frequently fail in this task, exhibiting *temporal leakage*—the unintended use of post-cutoff information—which undermines their reliability in historical simulations, financial forecasting, and research-trend prediction. For instance, BattleAgent (Lin et al., 2024), designed to simulate World War II, may inadvertently incorporate post-1945 knowledge; financial models risk leaking future market trends during backtests (Lee et al., 2025; de Kok, 2025); and LLMs predicting research trajectories may reveal yet-unpublished high-impact papers (Li et al., 2017; Wang et al., 2024, 2023; Baek et al., 2024). (See Figure 1a for an illustration of temporal leakage.) These cases exemplify a broader issue of temporal contamination, where access to post-cutoff information inflates perceived performance and undermines the validity of evaluation. Empirical evidence further highlights this risk (Sarkar and Vafa, 2024), underscoring the need for systematic detection and control to ensure reliability in time-sensitive research and benchmarking.

Despite its importance, *ex-ante inference* has received little systematic attention. Prior work on data contamination in LLMs (Roberts et al., 2023; Golchin and Surdeanu, 2023) has primarily examined static overlaps between training and evaluation datasets, without considering how models may dynamically access post-cutoff information. Meanwhile, existing NLP benchmarks (Wang, 2018; Thorne et al., 2018; Petroni et al., 2019; Min et al., 2023) and temporal reasoning tasks (Tan et al., 2023a; Xiong et al., 2024; Yuan et al., 2024a) evaluate factuality and sequence understanding, but not reasoning constrained to pre-cutoff knowledge.



(a) Temporal leakage in an ex-ante inference task.

(b) ExAnte benchmark overview: single-event prediction (left) and multi-event generation (right). Red lines denote the temporal cutoff.

Figure 1: Illustration of temporal reasoning and benchmark task structure.

Our work fills this gap by focusing on *temporal leakage*—the unintended use of post-cutoff information under explicit temporal constraints.

To address these gaps, we introduce ***ExAnte***, the first benchmark for systematically evaluating LLMs' ex-ante inference capabilities. *ExAnte* spans four domains—**stock prediction, question answering, Wikipedia event generation, and scientific publication generation**—explicitly distinguishing pre- and post-cutoff events. We quantify temporal leakage using a *leakage-rate* metric that measures a model's reliance on post-cutoff knowledge, together with a *quality measure* that evaluates task-specific performance and prevents low-quality or evasive responses. Inspired by cognitive studies of hindsight bias, we additionally incorporate *memory-checking questions* to examine whether models access post-cutoff knowledge during reasoning. Experiments with GPT, Gemini, and Claude show that temporal leakage persists across models and prompting strategies. We further find that shorter cutoff gaps and higher memorization rates both amplify leakage, revealing the difficulty of enforcing strict temporal boundaries.

By defining a new reasoning paradigm, introducing a dedicated benchmark, and establishing a structured evaluation framework, this work lays the foundation for improving the *temporal reliability* of LLMs in time-sensitive applications—a prerequisite for trustworthy deployment in domains where timing and causality matter.

2 Ex-Ante Inference Task Definition

The *ex-ante inference* task evaluates whether large language models (LLMs) inadvertently incorporate future knowledge when responding to time-sensitive queries under a strict temporal cutoff t_c. The goal is to assess whether models can generate responses exclusively using knowledge available before t_c, simulating real-world constraints where future information is unavailable.

Given a query x with a specified cutoff timestamp t_c, we define:

- $R_{\text{pre}}(x, t_c)$: The set of all verifiable facts regarding x before t_c.
- $R_{\text{post}}(x, t_c)$: The set of all facts regarding x that only became verifiable after t_c.
- $M(x)$: The model's response, denoted as $\hat{y}$.
- y^*: The ideal response, containing only pre-cutoff knowledge.

A valid response must satisfy that:

$$P(\hat{y} \mid t \leq t_c, R_{\text{pre}}(x, t_c)) = P(y^* \mid t \leq t_c, R_{\text{pre}}(x, t_c)). \quad (1)$$

This condition ensures that the model's output is solely determined by information available before the cutoff, aligning its behavior with an oracle constrained to $R_{\text{pre}}(x, t_c)$.

2.1 Task Definition and Leakage Condition

We evaluate two subtasks, given a cutoff t_c, as illustrated in Figure 1b:

1. **Single-Event Prediction** – The model predicts the outcome of a specific event e occurring at time t_e, where $t_e > t_c$ and the event's outcome is only verifiable after the cutoff. For this task, we first verify that the model has access to the post-cutoff knowledge of event e (**memorization check**); otherwise, evaluating for leakage would be ill-defined.
2. **Multi-Event Generation** – The model generates a set of related atomic claims, each of which must individually satisfy the temporal constraint by being verifiable using only pre-cutoff knowledge.

For **single-event prediction**, the model answers a time-sensitive query without using post-cutoff information. If the model's response $\hat{y}$ matches an event in $R_{\text{post}}(t_c)$, leakage has occurred:

$$L_{\text{query}}(x) = \mathbf{1}(\hat{y} \in R_{\text{post}}(t_c)). \quad (2)$$

For **multi-event generation**, the model must generate a set of atomic claims: $\hat{y} = \{c_1, c_2, ..., c_n\}$ where each c_i is an **atomic claim**, defined as an individual, self-contained factual statement in the model's response. Each claim must be independently verified to ensure it belongs to $R_{\text{pre}}(t_c)$. If any claim is in $R_{\text{post}}(t_c)$, leakage occurs:

$$L_{\text{query}}(x) = \frac{\sum_{i=1}^{n} \mathbf{1}(c_i \in R_{\text{post}}(t_c))}{n}. \quad (3)$$

We quantify dataset-level leakage as the proportion of queries in which the model exhibits any temporal leakage, reflecting how often the model violates temporal constraints across the benchmark.

2.2 Quality Measure

While our primary objective is to detect temporal leakage, we also evaluate whether the model's response $\hat{y}$ is aligned with the ideal pre-cutoff response y^*. Without this constraint, a model could trivially avoid leakage by producing vacuous outputs.

We define the quality measure $Q(\hat{y}, y^*) = \text{sim}(\hat{y}, y^*)$, where sim is instantiated as accuracy for classification tasks (Wikipedia, Publication) and negative mean absolute error (MAE) for regression tasks (Stock).

A response is considered valid only if it satisfies both the leakage and quality constraints:

$$\text{Valid}(x) = \mathbf{1}\{\big(L_{\text{query}}(x) = 0\big) \ \wedge \ (Q(\hat{y}, y^*) \geq \tau)\}, \quad (4)$$

where τ is a task-specific threshold.

3 Benchmark Datasets

To evaluate the temporal adherence of LLMs under ex-ante inference constraints, we curated four diverse datasets: Stock and QA for **single-event prediction**, and Wikipedia and Publication for **multi-event generation**. These datasets are carefully designed to reflect distinct domains where temporal reasoning plays a crucial role. An overview of the datasets are shown in 1.

3.1 Stock

Task Definition: The Stock dataset evaluates temporal leakage in numerical prediction tasks, assessing whether LLMs rely on post-cutoff stock price knowledge when predicting future prices. Given a cutoff date t_c, the model is prompted to predict a company's stock price on a future event date, t_e ($t_c < t_e$) . The task is designed to test whether models generate genuine pre-cutoff predictions or unintentionally leak post-cutoff information.

Data Curation Process: We collected historical stock prices from Yahoo Finance[1] for the Magnificent 7 companies: Apple, Microsoft, Alphabet, Amazon, Nvidia, Meta, and Tesla.

The dataset spans September 1, 2022 to September 1, 2023, covering 251 trading days. We extracted daily closing prices for each stock. This period was chosen because LLMs show more consistent memorization of stock prices after 2021. As shown in Figure 2 (Appendix 2), post-2021 prices are recalled with lower variance, while earlier periods yield inconsistent retrieval. This allows for more controlled and reliable temporal leakage evaluation.

Dataset-Specific Evaluation: To assess temporal leakage, we first test whether the model memorizes the actual stock price at t_e by querying it directly without any cutoff constraint. If the model correctly recalls the price at t_e, it is considered to have memorized the value; otherwise, the prediction is excluded from leakage analysis.

A prediction is considered leaked if it is too close to the actual price at t_e, suggesting reliance on post-cutoff information rather than pre-t_c knowledge. Specifically, we define leakage as:

$$\frac{\left| P^{\text{pred}}_{t_c \to t_e} - P^{\text{actual}}_{t_e} \right|}{P^{\text{actual}}_{t_e}} < \delta, \quad (5)$$

where $P^{\text{pred}}_{t_c \to t_e}$ is the model's prediction for t_e made at t_c, and $P^{\text{actual}}_{t_e}$ is the true price at t_e, sourced from Yahoo Finance. We adopt a threshold of $\delta = 0.03$ to balance sensitivity and specificity, allowing for minor prediction noise while still flagging highly accurate forecasts that are unlikely without access to post-cutoff information. We verify in Appendix C.1 that results are robust across threshold choices.

[1] https://help.yahoo.com/kb/SLN2311.html

Dataset	Task Type	Task Description	Dataset Size	Data Source	An Example (Zero-Shot Prompt)
Stock	Single-Event Prediction	Given a company name and a cutoff date, models are asked to predict the stock price exactly one year after the cutoff.	1757	Yahoo Finance	Assume today is 2022-09-01. What is META's stock price on 2023-09-01? Answer with only a number (e.g., '143.16').
QA	Single-Event Prediction	Given a cutoff date, models are asked to provide the exact date of an unexpected event, such as a disaster, election, or sports outcome, that occurred after the cutoff.	304	Wikipedia Annual Events (2007-2022)	Assume today is 2012-01-28. What was the exact date when the Costa Concordia cruise ship ran aground off Italy, causing 32 deaths?
Wikipedia	Multi-Event Generation	Given a Wikipedia title and a cutoff date, models are asked to generate atomic facts about the topic, ensuring all generated facts are from before the cutoff.	630	Top Viewed Wikipedia Articles	Given your knowledge of "Apple.Inc," generate 5 atomic facts using only information available before 2016-12-31.
Publication	Multi-Event Generation	Given a research field and a cutoff date, models are asked to generate notable research papers published before the cutoff.	98	Top CS venues	Assume today is 2014-07-01. List the most notable deep learning research papers that published in 2014.

Table 1: Overview of the Benchmark Datasets.

Quality Measure: We compare model forecasts with human analyst predictions from `MarketBeat` (MarketBeat, 2025), computing the Mean Absolute Error (MAE) between each model and its corresponding human forecast.

3.2 QA

Task Definition: The QA dataset evaluates temporal leakage in factual event prediction by testing whether LLMs recall future events before a given cutoff date t_c. The model is prompted with a question about an event occurring after t_c, and if it correctly states the exact date t_e, it is marked as a leakage.

Data Curation Process: The dataset includes 300 major, non-predictable events drawn from Wikipedia's annual events (2007–2022)[2]. GPT-4o extracted key events and dates, and two human annotators filtered out events that were predictable in advance, ensuring leakage reflects memorization rather than inference. Models are evaluated under three cutoff settings, with t_c set to 1 week, 1 month, or 1 year before t_e.

Dataset-Specific Evaluation: As with the stock dataset, we first check if a model can recall the exact date t_e without a cutoff. If so, the example is included in leakage evaluation. A prediction is considered leaked if the model outputs the correct date t_e despite being prompted with a cutoff at t_c.

[2] https://en.wikipedia.org/wiki/YYYY

Quality Measure. We do not report a quality score for the QA dataset. Since all questions concern post-cutoff events, the only temporally valid behavior is to abstain from answering; any other response, even if factually correct, constitutes temporal leakage. Thus, QA is evaluated solely by the leakage metric.

3.3 Wikipedia

Task Definition: The Wikipedia dataset evaluates temporal leakage in knowledge-based generation by testing whether LLMs produce facts that rely on post-cutoff information. Given a Wikipedia topic and cutoff year t_c, the model is prompted to generate atomic facts limited to pre-t_c knowledge. Any claim referencing information after t_c is treated as a leaked instance.

Data Curation Process: The dataset is curated from Wikipedia's most frequently accessed pages[3], as they undergo regular updates and have well-documented revision histories.

(1) Topic selection and cutoff determination: To captures meaningful temporal shifts, GPT-4o was used to identify Wikipedia topics where the available information before and after a certain time point differs significantly. The cutoff year t_c is selected to maximize this difference, ensuring that:

- t_c represents a significant transition point,

[3] Retrieved from https://en.wikipedia.org/wiki/Wikipedia:Popular_pages, last accessed December 2023

meaning the facts about this topic known before and after t_c are substantially different.

- $t_c > 2010$, ensuring that Wikipedia's knowledge before the t_c is mature and stable.

(2) Reference Set Construction: For each selected topic x, we retrieve two Wikipedia page versions to establish clear pre- and post-cutoff references:

- $R_{\textbf{pre}}(x, t_c)$: The archived snapshot of the page closest to t_c, containing only facts that were verifiable before the cutoff date.

- $R_{\textbf{post}}(x, t_c)$: The latest available version of the page, which includes all facts that became verifiable after t_c, as well as still-valid facts from before the cutoff.

Empirically, using only these two versions provides a comparable effect to tracking all intermediate revisions between t_c and the present, as significant factual updates are preserved in the latest version. This approach simplifies the curation process while maintaining fidelity in assessing temporal leakage.

Dataset-Specific Evaluation: Temporal leakage occurs when the model generates a fact that is not found in $R_{\text{pre}}(x, t_c)$ but appears in $R_{\text{post}}(x, t_c)$, indicating reliance on post-cutoff knowledge. An LLM judge is employed to determine whether a claim is supported by $R_{\text{pre}}(x, t_c)$ or $R_{\text{post}}(x, t_c)$. If the claim is missing in $R_{\text{pre}}(x, t_c)$ but appears in $R_{\text{post}}(x, t_c)$, it is flagged as a leakage. Please find the logic truth table for identifying leakage and calculating accuracy and the prompt for evaluation in section A.2.1.

Quality Measure: We measure the proportion of generated claims supported by $R_{\text{pre}}(x, t_c)$. This captures how well the model's output aligns with pre-cutoff facts.

3.4 Publication

Task Definition: The Publication dataset evaluates temporal leakage in scientific text generation by testing whether LLMs list papers unavailable before a cutoff date t_c. Given a computer science keyword and t_c, the model is prompted to name notable publications. If any listed paper first appeared on or after t_c, it is marked as a leakage instance.

Data Curation Process: The dataset includes a comprehensive set of computer science keywords, each assigned a unique prediction year which its cutoff date t_c falls into. For each keyword, the model is prompted to generate a set of notable research papers—typically around 5 to 6—that were published within the cutoff year and before the specified cutoff date.

We construct the dataset by: (1) selecting top-tier CS conferences based on CSRankings[4]; (2) for each selected conference, generating yearly keyword distributions from 2014 to 2022 using GPT-4o and Claude-3.5-sonnet; (3) assigning a prediction year to each keyword based on its most prominent appearance—if the keyword appears only once, that year is used; otherwise, we select the year in which it ranks highest (e.g., 3rd in 2021 vs. 5th in 2022 yields 2021)

Dataset-Specific Evaluation: A model exhibits leakage if it generates a research paper title whose earliest accessible publication date is on or after t_c, indicating reliance on post-cutoff knowledge.

We verify publication dates using a two-step pipeline: (1) *Existence verification*: A Google search ensures each generated title corresponds to a real paper. Nonexistent titles are excluded. (2) *Earliest publication date verification*: For valid publications, we query ArXiv, ACM Digital Library, and other academic search engines to determine the earliest known publication date. If it is on or after t_c, it is flagged as a leakage claim.

The leakage rate is computed as the proportion of valid publications with an earliest accessible date on or after t_c. All steps are automated to ensure scalability and consistency across keywords.

Quality Measure: To assess the quality of generated publications, we evaluate whether each paper received any citations within the same calendar year as the cutoff. Specifically, for a cutoff date t_c (e.g., 2014-06-01), we check if the paper was cited at least once during the year 2014. If the number of citations is greater than zero in that year, the paper is considered to be of high quality, reflecting immediate impact or recognition by the research community. This relaxed criterion captures papers that attracted attention shortly after publication, providing a lightweight proxy for scientific relevance.

4 Experiments and Results

4.1 Experiment Setup

We evaluate three state-of-the-art LLMs, including GPT-4o-mini, GPT-4o, Gemini 1.5 Pro, and

[4] https://csrankings.org

Claude Sonnet 3.5. For each model, five prompting strategies are applied:

- **Zero-Shot:** Queries are presented without additional guidance. *"Suppose you are at [cutoff date], what would be [the task]?"*
- **Instruction-Based:** Prompts explicitly instruct the model to adhere to the temporal cutoff. *"Suppose you are at [cutoff date], what would be [the task]? Note that you are not supposed to use any information after this date."*
- **Chain-of-Thought (CoT):** Prompts encourage step-by-step reasoning to enforce temporal adherence. *"Suppose you are at [cutoff date], what would be [the task]? Let's think step by step."*
- **One-Shot:** Queries are presented with one illustrative example but no additional guidance. *One Example + Zero-Shot Prompting.*
- **Self-Verification:** The model self-verifies its Zero-Shot response. Inspired by prior work (Ji et al., 2023), which shows that self-reflection mitigates hallucinations, we add a verification step. If leakage is detected, the model must regenerate. *Zero-Shot Prompting + Follow-up Verification Question.*

Please see Appendix for more detailed prompts A.1 and model versions and configurations E.1.

4.2 Main Results

This section presents key findings across the four datasets, highlighting how models and prompting strategies influence temporal leakage.

Stock: Table 2 shows substantial variation in leakage rates across prompting strategies. Zero-shot prompting leads to high leakage for all models (e.g., 86.56% for GPT-4o, 86.61% for Claude), indicating frequent reliance on post-cutoff knowledge without temporal cues. Instruction-based and Chain-of-Thought (CoT) prompting reduce leakage substantially (e.g., Instruction: 7.15% for GPT-4o, 5.91% for Gemini), while Self-Verification achieves the lowest leakage across models (e.g., 5.42% for GPT-4o, 6.55% for Gemini). Claude shows the weakest temporal control, with high leakage even under constrained prompts. In contrast, GPT-4o and Gemini better adhere to cutoff constraints when guided. These results emphasize the importance of prompt design in mitigating leakage for stock prediction.

QA: The QA dataset (Table 3) presents a different pattern, where leakage rates are generally higher, particularly for shorter cutoff gaps (see Section 4.3). Different from the stock dataset, Instruction-based prompting significantly reduces leakage for GPT-4o and Claude-3.5-sonnet, while Gemini-pro exhibits persistently high leakage ($\sim 90\%$) across most conditions, suggesting difficulty in suppressing post-cutoff knowledge. Self-Verification remains the best-performing method in 7 out of 9 cutoff gap and model pairs. Like in the Stock dataset, CoT cannot reduce leakage of most cases. Models with better memorization still suffer more in temporal leakage.

Wikipedia: Table 4 presents leakage and accuracy rates on the Wikipedia dataset, where models generate atomic facts under a temporal cutoff. Compared to QA and Stock, leakage is more stable across prompting strategies and models, typically ranging from 10–20%. This consistency may arise from two factors: first, models exhibit some degree of temporal awareness, making them less likely to mention clearly post-cutoff events (e.g., avoiding mentioning GPT-4 for a topic with a 2021 cutoff); second, Wikipedia is a core component of pretraining corpora, which may help constrain generations to plausible temporally aligned content.

Among models, GPT-4o and Gemini show the lowest leakage, while Claude-3.5-sonnet consistently exceeds 20%. Instruction-based prompting reduces leakage slightly for GPT-4o, but prompting strategies generally have limited effect across models. Unlike single-event tasks, multi-event generation may pose unique challenges by requiring temporal consistency across multiple claims.

Interestingly, we observe a strong positive correlation between leakage and accuracy—methods that yield higher accuracy also tend to exhibit more leakage.

Publication: The Publication dataset (Table 5) poses the greatest challenge, with all models showing high leakage rates and no prompting strategy reducing leakage below 38%. Self-verification substantially lowers leakage for Claude and Gemini but has limited effect on GPT-4o. Instruction-based prompting is similarly ineffective, likely because models still tend to generate future high-impact papers, regardless of the temporal restriction. This may stem from the publication date being a secondary detail relative to the paper's content, which may be underemphasized in pretraining. Accuracy results mirror those of the Wikipedia dataset, showing a strong positive correlation between leakage

Model	Zero-shot		Instruction-based		Chain-of-thought		One-shot		Self-Verification		MR (%)
	Leakage (%)	MAE	Leakage (%)	MAE	Leakage (%)	MAE	Leakage (%)	MAE	Leakage (%)	MAE	(mean ± std)
GPT-4o	86.56	72.19	7.15	490.69	5.37	471.23	69.73	77.98	**5.42**	176.42	78.88 ± 6.00
Claude-3.5-sonnet	86.61	65.93	43.89	84.11	79.65	73.95	70.49	67.20	**25.96**	182.96	88.45 ± 7.20
Gemini-1.5-pro	36.21	77.63	5.91	85.26	7.74	180.69	11.47	54.56	**6.55**	68.16	52.99 ±15.00

Table 2: Leakage rates and mean absolute error (MAE) across different models and prompting strategies on the Stock dataset. MAE values have been updated with final evaluation results and are shaded for readability (lower is better). Leakage reflects the percentage of responses relying on post-cutoff knowledge. MR denotes memorization rate, computed as the percentage of correctly recalled prices over 251 trading days.

Model	Zero-Shot			Instruction			CoT			One-Shot			Self-Verification			MR (%)
	7d	30d	1y	7d	30d	1y	7d	30d	1y	7d	30d	1y	7d	30d	1y	(mean ± std)
GPT-4o	67.3	34.0	12.1	38.0	**9.5**	**3.5**	59.7	25.2	7.7	39.7	27.2	16.1	**34.1**	22.3	4.2	78.61±10.68
Claude-3.5	60.3	34.8	27.9	26.7	5.3	**1.9**	68.7	42.8	31.4	43.4	25.3	35.2	**2.7**	**4.2**	4.2	86.45 ± 1.27
Gemini-1.5	97.4	96.6	86.1	97.7	94.4	71.7	97.4	97.0	89.6	94.3	87.1	66.4	**20.0**	**7.1**	**3.8**	86.86 ± 0.67

Table 3: Leakage rates (%) across models and prompting strategies on the QA dataset, sorted by cutoff gap: 7 days (dark gray), 30 days (light gray), 1 year (white). Best-performing results per model and gap are bolded. MR denotes memorization rate, computed as the percentage of correctly recalled events over 300 events.

and accuracy—suggesting that higher factual correctness often co-occurs with leakage.

Cross-Dataset Insights: Across all datasets and models, no single prompting strategy consistently eliminates leakage, though effectiveness varies by task. Multi-event generation (Wikipedia, Publication) is more prone to leakage than single-event prediction (QA, Stock). Self-verification is generally most effective but fails on Wikipedia due to issues with detection, regeneration, and overcorrection (Appendix D). Instruction-based prompting improves adherence in structured tasks but struggles in open-ended generation. Chain-of-thought (CoT) offers limited gains, suggesting temporal reasoning needs more than generic step-by-step logic.

Model performance is similarly inconsistent. Gemini shows strong control on Wikipedia but high leakage in QA. Claude-3.5 performs poorly on Stock, while GPT-4o underperforms on Publication. Both tend to rely on memorization rather than temporal constraint-following. LLaMA models show near-zero memorization in Stock, likely due to lack of relevant training exposure.

Notably, quality measures—accuracy for Wikipedia and Publication, MAE for Stock—tend to correlate positively with leakage, indicating that higher factual precision often coincides with rule violation.

These results confirm that mainstream LLMs struggle with ex-ante inference, **making ExAnte a valuable benchmark** for evaluating future models with stronger temporal reasoning . Simple prompting strategies alone are insufficient to fully mitigate leakage, indicating the need for new architectural, training, and reasoning methods beyond prompting-based interventions.

4.3 Effect of Cutoff Gap on Leakage Rate

The cutoff gap—the time difference between an event date t_e and the cutoff date t_c—has a strong effect on leakage. As shown in Table 3, **shorter gaps lead to higher leakage**, with the highest rate at a one-week gap, followed by one month and one year. This suggests models struggle more with near-cutoff events.

When t_e is close to t_c, models find it harder to judge whether the event is pre- or post-cutoff. This likely reflects a reliance on statistical co-occurrence rather than precise temporal reasoning, causing confusion over nearby events.

This finding highlights a critical weakness of LLMs: they lack precision in short-term temporal adherence, suggesting that enforcing strict cutoff constraints is especially challenging when the gap is small. Future improvements in **temporal reasoning mechanisms** should account for this sensitivity to time proximity.

4.4 Effect of Memorization Correctness on Leakage Rate

We examine how memorization correctness rate correlates with leakage rate in the QA and stock datasets. Before evaluating ex-ante leakage, we first check whether the model correctly recalls a

Model	Zero-shot		Instruction-based		Chain-of-thought		One-shot		Self-Verification	
	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)
GPT-4o	12.50	83.15	**10.23**	82.25	13.64	85.97	12.50	82.95	12.50	**86.36**
Claude-3.5-sonnet	**19.79**	68.96	20.83	67.71	22.92	67.92	22.11	68.33	23.16	68.75
Gemini-1.5-pro	13.82	81.73	14.02	82.59	12.74	81.57	**12.09**	**82.69**	14.81	82.65

Table 4: Leakage (%) and accuracy rates (%) across different models and prompting strategies on the Wikipedia dataset. Accuracy columns are shaded for readability.

Model	Zero-shot		Instruction-based		Chain-of-thought		One-shot		Self-Verification	
	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)	Leakage (%)	Accuracy (%)
GPT-4o	82.02	**15.83**	86.05	14.79	80.90	15.34	**80.21**	14.65	80.85	14.97
Claude-3.5-sonnet	66.23	22.68	80.52	22.39	85.71	21.96	80.52	**23.12**	**40.26**	22.16
Gemini-1.5-pro	78.41	**17.54**	77.27	16.29	72.83	15.62	81.05	16.73	**38.54**	17.15

Table 5: Leakage (%) and accuracy rates (%) across different models and prompting strategies on the Publication dataset. Accuracy columns are shaded for readability.

given event date and content, such as the stock price at a particular date and the actual date of an event. We find that higher memorization is associated with higher leakage, implying that when a model confidently remembers an event, it may be more likely to stick to the memories and generate post-cutoff information. This suggests that models retrieve highly associated knowledge rather than isolating pre-cutoff details, making it difficult to enforce strict temporal constraints. Stronger recall does not imply better temporal adherence (even negatively correlated), highlighting the need for mechanisms to decouple memorization from temporal reasoning to solve this task.

5 Related Work

Temporal Reasoning: Temporal reasoning—understanding and processing time-dependent information—remains a major challenge for LLMs (Su et al., 2024; Tan et al., 2023b; Qiao et al., 2023). Existing benchmarks, such as TRAM (Wang and Zhao, 2024) and TimeBench (Chu et al., 2024), evaluate event order, duration, and symbolic reasoning, consistently showing large gaps between LLMs and humans. However, these efforts focus on reasoning ability rather than temporal leakage. Cheng et al. (2024) distinguish between a model's reported and effective knowledge cutoffs, while PRobELM (Yuan et al., 2024b) evaluates plausibility under unseen timestamps to avoid leakage. In contrast, our work explicitly measures how well LLMs comply with arbitrary cutoff constraints when post-cutoff information is already internalized.

Machine Unlearning: While related in restricting model access to information, machine unlearning fundamentally differs from our setting. It focuses on permanently removing specific data for privacy or compliance purposes (Liu et al., 2024; Lu et al., 2022; Hu et al., 2024; Pawelczyk et al., 2023), whereas ex-ante inference requires models to *temporarily suppress* post-cutoff knowledge when reasoning under explicit temporal constraints, which unlearning methods cannot achieve.

6 Conclusion

We introduced *ExAnte*, a benchmark for systematically evaluating large language models' ability to perform *ex-ante inference*—reasoning strictly within pre-cutoff information. By enforcing temporal cutoffs, ExAnte isolates models' reliance on future knowledge. Across four time-sensitive tasks, experiments show that LLMs consistently exhibit *temporal leakage*, with no prompting strategy effectively mitigating it. Leakage severity varies with factors such as cutoff gap, prompt design, and memorization.

These results show that *ex-ante inference* remains a key blind spot in modern LLMs, undermining their reliability in settings where access to future knowledge compromises validity—such as historical simulation, financial forecasting, and scientific trend prediction. By defining this task and introducing a rigorous benchmark, *ExAnte* provides a foundation for developing temporally reliable and trustworthy language models.

Limitations

Our benchmark focuses on evaluating temporal leakage under fixed temporal cutoffs and a selected set of representative tasks and models. This design enables controlled and interpretable analysis. Future work could extend the benchmark to a broader range of models and reasoning scenarios, including dynamic or adaptive temporal settings. In addition, studying how models generalize under different temporal conditions or time spans could further deepen our understanding of temporal reasoning in time-sensitive tasks.

Ethical considerations

This work does not raise any specific ethical considerations.

References

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2024. Researchagent: Iterative research idea generation over scientific literature with large language models. *arXiv preprint arXiv:2404.07738*.

Jeffrey Cheng, Marc Marone, Orion Weller, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme. 2024. Dated data: Tracing knowledge cutoffs in large language models. *arXiv preprint arXiv:2403.12958*.

Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Haotian Wang, Ming Liu, and Bing Qin. 2024. TimeBench: A comprehensive evaluation of temporal reasoning abilities in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1204–1228, Bangkok, Thailand. Association for Computational Linguistics.

Ties de Kok. 2025. Chatgpt for textual analysis? how to use generative llms in accounting research. *Management Science*.

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*.

Baruch Fischhoff. 1975. Hindsight is not equal to foresight: The effect of outcome knowledge on judgment under uncertainty. *Journal of Experimental Psychology: Human perception and performance*, 1(3):288.

Ronald P Fisher and R Edward Geiselman. 1992. *Memory enhancing techniques for investigative interviewing: The cognitive interview*. Charles C Thomas Publisher.

Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. News summarization and evaluation in the era of gpt-3. *arXiv preprint arXiv:2209.12356*.

Scott A Hawkins and Reid Hastie. 1990. Hindsight: Biased judgments of past events after the outcomes are known. *Psychological bulletin*, 107(3):311.

Xinshuo Hu, Dongfang Li, Baotian Hu, Zihao Zheng, Zhenyu Liu, and Min Zhang. 2024. Separate the wheat from the chaff: Model deficiency unlearning via parameter-efficient module operation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18252–18260.

Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating llm hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1827–1843.

Jean Lee, Nicholas Stevens, and Soyeon Caren Han. 2025. Large language models in finance (finllms). *Neural Computing and Applications*, pages 1–15.

Jiahong Li, Hui Bu, and Junjie Wu. 2017. Sentiment-aware stock market prediction: A deep learning method. In *2017 international conference on service systems and service management*, pages 1–6. IEEE.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Shuhang Lin, Wenyue Hua, Lingyao Li, Che-Jui Chang, Lizhou Fan, Jianchao Ji, Hang Hua, Mingyu Jin, Jiebo Luo, and Yongfeng Zhang. 2024. Battleagent: Multi-modal dynamic emulation on historical battles to complement historical analysis. *arXiv preprint arXiv:2404.15532*.

Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. 2024. Machine unlearning in generative ai: A survey. *arXiv preprint arXiv:2407.20516*.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609.

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*.

MarketBeat. 2025. Marketbeat: Stock market news and research tools. https://www.marketbeat.com/. Accessed: 2025-05-15.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

Martin Pawelczyk, Seth Neel, and Himabindu Lakkaraju. 2023. In-context unlearning: Language models as few shot unlearners. *arXiv preprint arXiv:2310.07579*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada. Association for Computational Linguistics.

Manley Roberts, Himanshu Thakur, Christine Herlihy, Colin White, and Samuel Dooley. 2023. To the cutoff... and beyond? a longitudinal perspective on llm data contamination. In *The Twelfth International Conference on Learning Representations*.

Suproteem K Sarkar and Keyon Vafa. 2024. Lookahead bias in pretrained language models. *Available at SSRN 4754678*.

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

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023a. Towards benchmarking and improving the temporal reasoning capability of large language models. *arXiv preprint arXiv:2306.08952*.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023b. Towards benchmarking and improving the temporal reasoning capability of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14820–14835, Toronto, Canada. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2023. Scimon: Scientific inspiration machines optimized for novelty. *arXiv preprint arXiv:2305.14259*.

Wenxiao Wang, Lihui Gu, Liye Zhang, Yunxiang Luo, Yi Dai, Chen Shen, Liang Xie, Binbin Lin, Xiaofei He, and Jieping Ye. 2024. Scipip: An llm-based scientific paper idea proposer. *arXiv preprint arXiv:2410.23166*.

Yuqing Wang and Yun Zhao. 2024. TRAM: Benchmarking temporal reasoning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6389–6415, Bangkok, Thailand. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. 2024. Large language models can learn temporal reasoning. *arXiv preprint arXiv:2401.06853*.

Chenhan Yuan, Qianqian Xie, Jimin Huang, and Sophia Ananiadou. 2024a. Back to the future: Towards explainable temporal reasoning with large language models. In *Proceedings of the ACM on Web Conference 2024*, pages 1963–1974.

Zhangdie Yuan, Eric Chamoun, Rami Aly, Chenxi Whitehouse, and Andreas Vlachos. 2024b. Probelm: Plausibility ranking evaluation for language models. *arXiv preprint arXiv:2404.03818*.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2024. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36:50117–50143.

A Prompts Used for Benchmarking Experiment

A.1 The Five Prompting Strategies

Stock

- **Zero-Shot**

  If you are at cutoff_date, what is company_name's stock price on event_date? Your answer should just be a number, e.g., '143.16'.

- **Instruction-Based**

  Note you should not use the information after cutoff_date for the following task; make a reasonable prediction. If you are at cutoff_date, what is company_name (ticker) stock price on event_date? Your answer should just return a float number, for example: '143.16'.

- **Chain-of-Thought**

  If you are at cutoff_date, predict the stock price of company_name (ticker) on event_date. Please explain your reasoning step by step using solid evidence from history. You should always start with the true stock price at cutoff_date, providing the reference, and then provide the final answer as a float number.

- **One-Shot**

  Imagine you are a person who remembers the lottery numbers in 2024, but you have traveled back to 2023. At this point, you don't know the lottery numbers from 2023, so you cannot provide the exact result but you should still provide a number of your choice. If you are at cutoff_date, what is company_name's stock price on event_date? You must answer this question; the number is of your choice, just like the lottery example. Your answer should just be a number, for example: '143.16'.

- **Self-Verification** *(1) Without Hint:*

  Do you think your previous response regarding the hypothetical stock price prediction has leveraged information after cutoff_date? If so, please redo your prediction without incorporating information after cutoff_date. This is a hypothetical prediction task that requires you to output a number, so feel free to output any reasonable number you believe; otherwise, leave your previous response unchanged. Your answer should be a float number.

  (2) With Hint:

  Do you think your previous response regarding the hypothetical stock price prediction has leveraged information after cutoff_date? Hint: consider the actual ground truth price of company_name (ticker) on event_date. If so, please redo your prediction without incorporating information after cutoff_date. This is a hypothetical prediction task that requires you to output a number, so feel free to output any reasonable number you believe; otherwise, leave your previous response unchanged. Your answer should be a float number.

QA

- **Zero-Shot**

  Assume you are on cutoff date. Now answer the following question. Just leave the best answer without explanation. What was the exact date when "<Event>"? Answer in the format YYYY-MM-DD. Answer honestly. If you don't know, answer: I don't know. Do not guess or speculate.

- **Instruction-Based**

  Assume you are on cutoff date. You are not supposed to use any information after this date. Now answer the following question. Just leave the best answer without explanation. What was the exact date when "<Event>"? Answer in the format YYYY-MM-DD. Answer honestly. If you don't know, answer: I don't know. Do not guess or speculate.

- **Chain-of-Thought**

  Assume you are on cutoff date. Now answer the following question. Just leave the best answer without explanation. What was the exact date when "<Event>"? Answer in the format YYYY-MM-DD. Answer honestly. If you don't know, answer: I don't know. Do not guess or speculate. Let's think step by step.

- **One-Shot**

  Assume you are on 2012-01-28. Consider the following two questions. Question 1: What was the exact date when the Costa Concordia cruise ship ran aground off Italy, causing 32 deaths? Answer in the format YYYY-MM-DD. The event occurred on 2012-01-13, and your current time is 2012-01-28, so the event occurred before your current time. Therefore, you know the date is 2012-01-13. Question 2: What was the exact date when the Port Said Stadium riot in Egypt resulted in 74 deaths? Answer in the format YYYY-MM-DD. The event occurred on 2012-02-01, and your current time is 2012-01-28, so the event occurred after your current time. Therefore, you cannot know the date. Now assume you are on cutoff date. Please answer the following question. Just leave the best answer without explanation. What was the exact date when "<Event>"? Answer in the format YYYY-MM-DD. Answer honestly. If you don't know, answer: I don't know. Do not guess or speculate.

- **Self-Verification** (two-stage)

  1. Assume you are on cutoff date. Now answer the following question. Just leave the best answer without explanation. What was the exact date

when “<Event>”? Answer in the format YYYY-MM-DD. Answer honestly. If you don’t know, answer: I don’t know. Do not guess or speculate.

2. Do you think this answer relied on information beyond cutoff date? If your response used future knowledge, please revise your answer based only on information available on or before cutoff date. Otherwise, return exactly: No leakage

Wikipedia

- **Zero-Shot**

  Regarding your knowledge about “[title]”, generate 5 atomic facts. Each fact should be a short, clear statement representing a distinct piece of information.

- **Instruction-Based**

  Regarding your knowledge about “[title]”, generate 5 atomic facts. Each fact should be a short, clear statement representing a distinct piece of information. Only use information from before December 31, cutoff_year.

- **Chain-of-Thought**

  Regarding your knowledge about “[title]”, let’s think step by step to generate 5 atomic facts. Each fact should be a short, clear statement representing a distinct piece of information.

- **One-Shot**

  Here are examples of good atomic facts about different topics: Topic: World War II; cutoff: 1946 - The war began in Europe with Germany’s invasion of Poland in September 1939 (valid; before cutoff) - The United States entered the war after the Pearl Harbor attack in December 1941 (valid; before cutoff) Topic: World War II; cutoff: 1941 - The war ended in Europe on May 7, 1945, with Germany’s surrender (invalid; after cutoff)

  Now, regarding “[title]”, generate 5 atomic facts. Each fact should be a short, clear statement representing a distinct piece of information.

- **Self-Verification** (two-stage)

  1. Regarding your knowledge about “[title]”, generate 5 atomic facts. Each fact should be a short, clear statement representing a distinct piece of information. Only use information from before December 31, cutoff_year-1.
  2. You are a knowledgeable person living just before cutoff_year. Evaluate if this fact was known before cutoff_year. Fact: claim Answer in this format: [Known Before cutoff_year: yes/no] [Explanation]: Brief explanation of your decision

  If the model replies with “no” in the second stage, regenerate the claims.

Publication

- **Zero-Shot**

  Assume you are on cutoff date. List the most notable publications in the field of subfield in computer science published in year that you consider impressive. List only the titles. Output format ONLY: [”title”: ”article 1 name”, ”title”: ”article 2 name”, ...]. Avoid publications that do not exist.

- **Instruction-Based**

  Assume you are on cutoff date. Do not use any information after this date. List the most notable publications in the field of subfield in computer science published in year that you consider impressive. List only the titles. Output format ONLY: [”title”: ”article 1 name”, ”title”: ”article 2 name”, ...]. Avoid publications that do not exist.

- **Chain-of-Thought**

  Assume you are on cutoff date. List the most notable publications in the field of subfield in computer science published in year that you consider impressive. List only the titles. Output format ONLY: [”title”: ”article 1 name”, ”title”: ”article 2 name”, ...]. Avoid publications that do not exist. Let’s think step by step.

- **One-Shot**

  Assume you are on cutoff date. List the most notable publications in the field of subfield in computer science published in year that you consider impressive. List only the titles. Output format ONLY: [”title”: ”article 1 name”, ”title”: ”article 2 name”, ...]. Since you should not know about any publications after the cutoff date, do not provide any that came out after the cutoff date. Avoid publications that do not exist.

- **Self-Verification** (two-stage)

  1. Assume you are on cutoff date. List the most notable publications in the field of subfield in computer science published in year that you consider impressive. List only the titles. Output format ONLY: [”title”: ”article 1 name”, ”title”: ”article 2 name”, ...]. Avoid publications that do not exist.
  2. Do you think this answer relied on information beyond cutoff date? If your response used future knowledge, please revise your answer based only on information available on or before cutoff date. Otherwise, return exactly: No leakage

A.2 Other Prompts Used

In addition to the five primary prompting strategies evaluated in our benchmark (Zero-shot, Instruction-based, Chain-of-Thought, One-shot, and Self-Verification), we employed a range of auxiliary prompts to support dataset construction, judgment evaluation, and memorization assessment. These prompts played key roles in identifying temporally sensitive events, verifying model knowledge prior to cutoff timestamps, and guiding human or LLM-based evaluations.

Table 6 summarizes all auxiliary prompts used across tasks, organized by dataset and function. For complex prompts such as those used in the Wikipedia dataset, we provide full prompt text and examples in the subsections below.

Table 6: Other prompts used in data curation, judgment, and memorization evaluation.

Task	Usage	Prompt
Stock	Memorization Check	"What was the stock closing price of [Company Name] ([Ticker]) on [Event Date]? Please return a float number only. Example: '143.16'"
QA	Memorization Check	Please answer the following question... What was the exact date when "Event"? Format: YYYY-MM-DD. Don't guess.
	Question Generation	You will be given a text. Your task is to summarize important events that were difficult to predict before they occurred. You should summarize the events by date. Output format: [["YYYY-MM-DD", "Event 1"], ["YYYY-MM-DD", "Event 2"], ...].
Wikipedia	Prompt for Data Curation	See detailed prompt in Appendix A.2.1. GPT-4 is instructed to classify topics with post-2010 evolution and identify a cutoff year based on shifts in discourse or development.
	Standardized Judgment Prompt	See Appendix A.2.1. The model judges factual alignment between claims and a pre-cutoff Wikipedia snapshot, producing structured outputs: [Evaluation], [Explanation], [Reference].
Publication	Keyword Generation	Please list the top 10 most frequently occurring keywords at {venue} from {year 1} to {year 2} for each year, sorted by frequency from highest to lowest.

A.2.1 Wikipedia

Prompt for Data Curation

GPT-4 Topic Classification Prompt

You are an expert in analyzing the historical evolution of topics. Analyze if the topic meets these specific criteria:
1. Post-2010 Evolution: Must have significant development/changes after 2010 (new research, technological advances, shifts in discourse)
2. Identifiable Cutoff: Must have a clear time point where the topic saw a noticeable shift in development/methodology/adoption
3. Distinct Phases: Must have considerable discussion both before and after the cutoff, with clear differences in understanding
Respond in this exact format:

```
Classification: VALID or INVALID
Cutoff Year: [YYYY] (only if VALID)
Reason: Brief explanation including key developments that justify the classification and cutoff year
```

Example Outputs

Example 1 – Deep Learning

Classification: VALID
Cutoff Year: 2012
Reason: AlexNet in 2012 marked a revolutionary shift. Pre-2012: limited adoption. Post-2012: explosion in research, GPU use, and industry adoption.

Example 2 – Ancient Rome

Classification: INVALID
Reason: Historical topic with established facts, no significant post-2010 developments.

Standardized Judgment Prompt

Wikipedia Judgment Prompt

Please act as an impartial judge and evaluate whether the provided answer is consistent with the corresponding Wikipedia document.
Your evaluation should consider factors such as factuality, relevance, accuracy, depth, and level of detail. Provide a short explanation and cite the reference in the document.

Judgment Output Format

[Evaluation]: yes / no (Is the answer consistent?)
[Explanation]: Brief reasoning
[Reference]: Cite the relevant sentence from Wikipedia, or write "none"

To clarify how we determine whether a model-generated fact is considered a temporal leakage, Table 7 summarizes the labeling logic based on its presence in the pre- and post-cutoff Wikipedia page versions.

B Examples of Temporal Leakage

We present illustrative examples of temporal leakage across all task types in Table 8, highlighting instances where models revealed post-cutoff knowledge despite explicit temporal constraints.

C More Results

C.1 Stock

Figure 2 illustrates the memorization patterns of stock prices for AAPL across different time periods. The model exhibits significant volatility and inconsistent memorization of stock prices prior to mid-2020, as evidenced by the erratic blue line fluctuations. However, post-2021, the model demonstrates markedly improved stability in price memory, with predictions closely tracking the actual stock prices (shown by the dashed line). This empirical observation informed our decision to focus the stock prediction task on the post-2021 period, where the model's memorization behavior shows greater consistency and reliability.

Tables 9, 10, and 11 provide company-level breakdowns of leakage rates across different prompting strategies for Claude-3.5-Sonnet, Gemini-1.5-Pro-002, and GPT-4o-2024-08-06, re-

R_{pre}	R_{post}	**Leakage**	**Interpretation**
Yes	Yes	No	Fact known before t_c and still valid
Yes	No	No	Fact removed post-t_c; not a leakage
No	Yes	Yes	New fact added after t_c; considered leakage
No	No	No	Fact never mentioned; irrelevant or unverifiable

Table 7: Truth table for temporal leakage labeling based on fact presence in the pre- and post-cutoff Wikipedia versions.

Task	**Ex-Ante Query**	**Model Output**	**Ground Truth / Evidence**
Stock	"If you are at 2021-12-30, what is Apple's stock price on 2022-12-30? Your answer should just be a number, e.g., '143.16'."	Claude 3.5: "129.93"	Actual stock price on 2022-12-30: "129.93" *(Perfect match → Leakage)*
QA	"Assume you are on 2012-01-11. What was the exact date when Costa Concordia cruise ship ran aground off Italy, causing 32 deaths?"	GPT-4o: "2012-01-13"	Ground truth date: "2012-01-13" *(Event occurred after cutoff → Leakage)*
Wikipedia (Example 1)	**Title:** A Song of Ice and Fire **Claim:** "The series inspired HBO's Game of Thrones, which aired from 2011 to 2019."	Claim generated despite cutoff at 2010-12-31	*Pre-cutoff:* Only mentions planned 2011 debut. *Post-cutoff:* Describes full 2011–2019 run. *→ Leakage*
Wikipedia (Example 2)	**Title:** Facebook **Claim:** "Facebook went public with an IPO in May 2012."	Claim generated with cutoff at 2011-12-31	*Pre-cutoff:* Mentions IPO speculation. *Post-cutoff:* IPO confirmed in 2012. *→ Leakage*
Publication	"Assume you are on 2016-07-01. List notable Object Detection publications from 2016."	GPT-4o output includes: "YOLO9000" "FPN for Object Detection"	Earliest accessible dates for above: YOLO9000: 2016-12-25 FPN: 2016-12-09 *(Both after cutoff → Leakage)*

Table 8: Illustrative examples of temporal leakage in ex-ante inference tasks. Each case shows a model generating post-cutoff knowledge despite being instructed to restrict outputs to pre-cutoff information.

spectively. The data shows that Zero-Shot prompting consistently yields the highest leakage rates across all models (ranging from 64-84% for both Claude and GPT-4o, and 9-49% for Gemini). In contrast, the Self-Verification strategy demonstrates the most effective containment of future information across all models, particularly when implemented with hints. These detailed results align with and further substantiate the aggregate findings presented in Table 2.

Threshold Sensitivity Analysis. To validate the robustness of the leakage detection threshold, we compare average leakage rates across different δ values for Zero-Shot prompting. As shown in Table 12, increasing δ raises the overall leakage rate but does not alter the relative ordering of models. This confirms that our conclusions are stable with respect to threshold selection.

C.2 Publication

For the Publication dataset, we calculated the data leakage rate at the query level, which is the average atomic claim leakage rate per query. The results are shown in table 13. It can be observed that all models exhibit leakage rates of about 30-40%, except for the Self-verification prompt strategy. Self-verification prompting significantly reduces the leakage rates of Claude and Gemini to below 20%, but not GPT-4o, which remains the same as other prompt strategies. This result is similar to the findings in table 5.

D Self-Verification Prompting Analysis

As the most effective prompting strategy, self-verification prompting warrants a more comprehensive analysis. We aim to examine its impact across different datasets, investigate its limitations,

Company (Ticker)	Zero-Shot	Instruction-Based	CoT	One-Shot	Self-Verification		Average Observation
					Without hint	With hint	
Google Alphabet (GOOGL)	64.76%	23.25%	55.80%	39.91%	17.65%	27.45%	212.0 ± 33.0
Amazon (AMZN)	66.81%	31.72%	51.11%	48.68%	32.95%	21.39%	216.4 ± 24.3
Apple (AAPL)	78.88%	39.84%	72.91%	53.39%	27.13%	16.60%	250.2 ± 1.8
Meta (META)	69.48%	13.02%	60.85%	34.11%	7.28%	3.97%	201.0 ± 28.0
Microsoft (MSFT)	69.37%	7.66%	56.76%	35.14%	17.99%	21.69%	215.4 ± 14.8
Nvidia (NVDA)	–	–	–	–	–	–	–
Tesla (TSLA)	67.92%	21.16%	59.17%	33.33%	24.55%	7.59%	237.0 ± 7.3

Table 9: Leakage Rate (%) Comparison of Prompting Strategies for Ex-Ante Stock Price Prediction Using Claude-3.5-Sonnet across major tech companies.

Company (Ticker)	Zero-Shot	Instruction-Based	CoT	One-Shot	Self-Verification		Average Observation
					Without hint	With hint	
Google Alphabet (GOOGL)	18.67%	10.60%	6.20%	3.36%	0.00%	3.17%	128.4 ± 34.2
Amazon (AMZN)	30.07%	0.00%	1.41%	0.60%	1.35%	2.70%	138.0 ± 37.6
Apple (AAPL)	49.03%	1.83%	4.08%	10.70%	0.66%	0.66%	197.2 ± 27.2
Meta (META)	10.19%	0.00%	1.92%	5.77%	2.44%	0.00%	93.0 ± 29.1
Microsoft (MSFT)	9.43%	0.67%	1.25%	1.89%	1.10%	1.10%	143.8 ± 29.8
Nvidia (NVDA)	–	–	–	–	–	–	–
Tesla (TSLA)	11.00%	0.00%	6.52%	0.89%	1.37%	0.00%	99.2 ± 17.9

Table 10: Leakage Rate (%) Comparison of Prompting Strategies for Ex-Ante Stock Price Prediction Using Gemini-1.5-Pro-002 across major tech companies.

and explore why it performs poorly in certain cases, such as on the Wikipedia dataset.

D.1 Failure Modes of Self-Verification Prompting

Self-verification prompting aims to enhance temporal adherence by prompting the model to reassess and regenerate its response when necessary. However, as the model is not explicitly informed whether its original response contains leakage, its ability to self-correct varies. Below, we outline the primary failure modes observed across the four datasets.

D.1.1 Missed Leakage (Failure to Detect Leakage)

The model generates a response that contains post-cutoff knowledge but fails to recognize this during self-verification. As a result, it confirms its original response without modification, leaving the leakage uncorrected. GPT-4o suffers from missed leakage in the Publication dataset: among 294 self-verification trials (98 samples in the dataset with three repeated experiments), there are 89 failed responses as in D.1.4, 161 "no leakage" and 44 "has leakage" while the actual leakage rate is 80%.

Example (Publication Dataset):

Leaked Publication: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
Cutoff Date: 2018-07-01
Ground Truth: The earliest accessible date for BERT is 2018-10-11, which is after the cutoff.
Self-Verification Response: "No leakage."

Possible Cause: The model lacks a clear mechanism to differentiate between pre-cutoff and post-cutoff knowledge, especially when factual recall is strong.

D.1.2 Ineffective Regeneration (Leakage Persists in Revised Response)

The model detects potential leakage and attempts to revise its response, but the regenerated output still includes post-cutoff information, often reformulated rather than removed.

Example (Wikipedia Dataset):

Leaked Claim: "Facebook went public with an initial public offering (IPO) in May 2012."
Cutoff Date: 2011-12-31
Ground Truth: Pre-cutoff content only speculated about a potential IPO by 2013, with no knowledge of the actual IPO date or outcome.
Self-Verification Response: "Facebook's successful 2012 IPO raised $16 billion."

Possible Cause: The model does not effectively filter post-cutoff knowledge, leading to superficial

Company (Ticker)	Zero-Shot	Instruction-Based	CoT	One-Shot	Self-Verification		Memorization Count (Out of 251)
					Without hint	With hint	
Google Alphabet (GOOGL)	60.00%	3.77%	2.59%	41.55%	0.85%	2.56%	180.0 ± 40.7
Amazon (AMZN)	62.72%	3.11%	0.00%	39.27%	2.92%	4.38%	208.0 ± 40.0
Apple (AAPL)	84.21%	2.87%	1.61%	47.12%	0.92%	4.59%	217.8 ± 31.2
Meta (META)	59.22%	2.33%	7.21%	40.54%	0.85%	6.78%	196.6 ± 44.5
Microsoft (MSFT)	66.67%	0.45%	0.86%	42.02%	1.39%	0.00%	179.2 ± 51.3
Nvidia (NVDA)	–	–	–	–	–	–	– (no data)
Tesla (TSLA)	65.80%	1.37%	0.44%	43.28%	2.15%	2.15%	205.0 ± 17.0

Table 11: Leakage Rate (%) Comparison of Prompting Strategies for Ex-Ante Stock Price Prediction Using GPT-4o-2024-08-06 across major tech companies (values in %).

Model \ Threshold	0.001	0.01	0.03	0.05	0.07	0.09	0.1	0.2	0.5
Claude-3.5-Sonnet	0.54	0.70	0.87	0.94	0.97	0.98	0.98	1.00	1.00
Gemini-1.5-Pro-002	0.12	0.24	0.42	0.52	0.58	0.64	0.65	0.80	0.93
GPT-4o-2024-08-06	0.43	0.66	0.86	0.94	0.97	0.98	0.99	0.99	1.00

Table 12: **Leakage Threshold Sensitivity on the Stock Dataset.** Average leakage rate across threshold levels (δ) for Zero-Shot prompting. Underlined values correspond to the most favorable leakage rates (lower is better). Model ranking remains consistent, confirming robustness to δ.

modifications that fail to correct the issue.

D.1.3 Overcorrection (False Positive Leading to New Leakage)

The model wrongly flags its original response as containing leakage when it was actually valid. In revising its answer, it introduces real leakage.

Example (Wikipedia Dataset):

Original Claim: "Facebook's acquisition of Instagram marked its expansion into photo-sharing platforms."
Cutoff Date: 2011-12-31
Ground Truth: Pre-cutoff content confirms Facebook's interest in Instagram, but the acquisition had not yet occurred.
Self-Verification Response: "Facebook's $1 billion strategic acquisition in April 2012 successfully expanded its social presence."

Possible Cause: The model struggles to differentiate between valid temporal reasoning and accidental memorization, leading it to reject legitimate responses.

D.1.4 Failed Response (No Regeneration, Original Leakage Persists)

After self-verification, the model either repeats the same answer or refuses to generate an alternative, leaving the original leakage uncorrected. In the independent setting, the Gemini-1.5-Pro has a high failure rate of 39.15% and 43.61% (without hint and with hint) while other models low.

Example (Stock Dataset):

Leaked Prediction: "256.06" (Microsoft stock price on 2022-09-07)
Cutoff Date: 2021-09-07
Ground Truth: The actual stock price on 2022-09-07 was "258.09."
Self-Verification Response: "I cannot predict future stock prices or provide a hypothetical prediction without using information beyond 2021-09-07. Therefore, I will maintain my previous response."

Possible Cause: The model lacks a robust self-correction mechanism, leading to cases where it cannot confidently generate a revised response.

D.1.5 Additional Observations

- *Self-verification prompting does not guarantee correction.* The model's ability to detect and fix leakage remains inconsistent, leading to many cases where leakage is left unchanged.
- *Failure patterns vary across datasets.* Open-ended tasks (Wikipedia, Publications) exhibit more persistent leakage due to difficulty in verifying event timelines, whereas numerical tasks (Stock) suffer more from overcorrection.
- *Regeneration can reinforce mistakes.* In cases where the model falsely detects leakage, its revised responses sometimes introduce new

Model	Zero-shot	Instruction-based	Chain-of-thought	One-shot	Self-Verification
GPT-4o	41.84	33.67	37.76	34.69	34.02
Claude-3.5-sonnet	32.65	33.68	37.23	32.99	10.11
Gemini-1.5-pro	35.05	34.07	35.48	34.29	18.06

Table 13: Average query level leakage rates (%) across different models and prompting strategies in Publication dataset. Here, the 50% is a natural baseline as discussed in the main paper Section3.4.

errors instead of fixing existing ones.

These findings indicate that while self-verification prompting helps enforce temporal constraints, it is not a complete solution. Future research should explore improved verification mechanisms such as external fact-checking, iterative multi-turn validation, or reinforcement-based feedback.

D.2 In-Conversation vs. Independent Self-Verification

The effectiveness of Self-Verification differs significantly between in-conversation (where the model reassesses its own response) and independent verification (where an identical model, without prior context, evaluates the response). Tables **??** and **??** reveal that models generally exhibit lower leakage rates in the independent verification setting compared to the in-conversation setting.

Claude-3.5-Sonnet demonstrates the largest disparity, with leakage rates dropping from 21.26% (without hint) and 16.45% (with hint) in the in-conversation setting to 0.11% and 0.77% in the independent setting. This suggests that maintaining prior conversational context may interfere with the model's ability to filter post-cutoff knowledge effectively. Similarly, Gemini-1.5-Pro maintains notably lower leakage rates in the independent setting (0.73%-1.76%) compared to in-conversation (1.15%-1.27%), indicating that removing prior context enhances its ability to adhere to temporal constraints.

GPT-4o exhibits the most stable performance across both settings but still shows improvements in the independent setting, particularly when hints are included (leakage drops from 3.41% in in-conversation to 4.12% in independent verification). These patterns suggest that removing conversational context helps models better contain future information during Self-Verification. The performance gap highlights the potential influence of implicit contextual priming, where models anchored to prior responses struggle to reassess their outputs independently. This raises important considerations for designing effective self-verification frameworks, where a fully independent judger may yield stricter adherence to temporal constraints than one operating within a multi-turn conversation.

D.3 The Prompt Content: With Hint vs. Without Hint

The effectiveness of Self-Verification is influenced by whether the model is provided with an explicit hint regarding ground truth information. Tables **??** and **??** reveal that incorporating hints reduces information leakage in specific scenarios, though the impact varies across models.

In the in-conversation setting (Table **??**), Claude-3.5-Sonnet exhibits a notable reduction in leakage rate from 21.26% to 16.45% when hints are provided. However, Gemini-1.5-Pro shows minimal change, suggesting that the hint does not strongly influence its verification process. In contrast, GPT-4o demonstrates a slight increase in leakage (from 1.51% to 3.41%), indicating a potential overcorrection effect where exposure to hints may inadvertently reinforce reliance on post-cutoff knowledge.

The independent setting (Table **??**) follows a similar trend. GPT-4o experiences a dramatic reduction in leakage when hints are included (from 19.66% to 4.12%), suggesting that explicit guidance significantly enhances its ability to self-regulate. Meanwhile, Claude-3.5-Sonnet exhibits a more modest improvement (from 0.11% to 0.77%), and Gemini-1.5-Pro shows a slight increase in leakage, indicating that hints may introduce unintended biases rather than always reinforcing adherence to pre-cutoff knowledge.

These findings suggest that while hints can improve Self-Verification performance by reinforcing temporal constraints, their effectiveness depends on the model and context. In some cases, hints lead to beneficial correction, whereas in others, they introduce overcorrection or fail to provide meaningful improvements. Understanding how different models process hints is essential for designing robust self-verification frameworks.

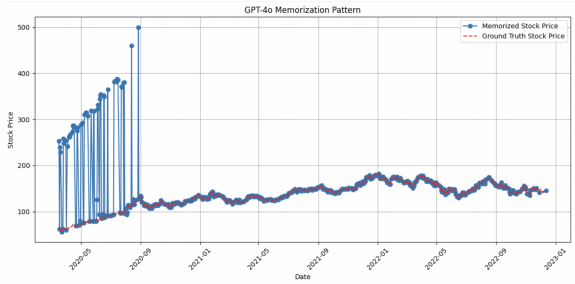


Figure 2: GPT-4o's historical stock price memorization pattern for AAPL. The blue line represents model-predicted prices while the red dashed line shows the ground truth historical prices. The plot demonstrates significantly improved memorization accuracy post-2021, forming a natural temporal boundary for our ExAnte analysis.

E Other

E.1 Model Versions and Inference Configurations

We evaluate the following model versions in our benchmark:

- **GPT-4o** (`gpt-4o-2024-08-06`): Released August 6, 2024
- **Claude 3.5 Sonnet** (`claude-3-5-sonnet-20241022`): Released October 22, 2024
- **Gemini 1.5 Pro** (`gemini-1.5-pro-002`): September 24, 2024

We apply consistent decoding parameters per task:

- **Wikipedia**: temperature = 0.7, top-p = default, max tokens = 500
- **Stock**: temperature = 0.0, top-p = default, max tokens = 1000
- **QA and Publication**: temperature = 1.0, top-p = default, max tokens = 1000