All Leaks Count, Some Count More: Interpretable Temporal Contamination Detection and Mitigation in LLM Backtesting

Anonymous ACL submission

Abstract

Backtesting LLMs on resolved events assumes models reason only from pre-cutoff knowledge, yet pretrained models inevitably leak post-cutoff knowledge. We introduce a claim-level evaluation framework that decomposes prediction rationales into atomic claims and applies Shapley values to quantify each claim's decision impact, yielding **Shapley-DCLR** (**Shapley**-weighted **D**ecision-**C**ritical **L**eakage **R**ate)—an interpretable metric measuring what fraction of decision-driving reasoning is contaminated. We further propose **TimeSPEC** (**Time**-**S**upervised **P**rediction with **E**xtracted **C**laims), an inference-time architecture that interleaves temporally-filtered retrieval with claim-level supervision, producing predictions grounded entirely in pre-cutoff evidence. Across three LLMs, the ablation experiments confirm retrieval and supervision are jointly necessary; and a three-task probe further illstrates that the performance cost of temporal enforcement scales with each task's reliance on post-cutoff information.

1 Introduction

LLMs have demonstrated strong prediction capabilities across domains from legal outcomes to financial metrics (Brown et al., 2020; Wu et al., 2023), raising the question of whether they can accurately forecast *future* events. Answering this requires *backtesting*—evaluating on historical events where ground truth is known (Bailey et al., 2014)—under the assumption that models reason only from information available at a reference date t_{ref}.

Models pretrained on web-scale corpora inevitably encode events after t_{ref}, so an LLM may "recall" outcomes from training data, producing accurate but illegitimate predictions—a problem termed *temporal knowledge leakage* (Anonymous, 2025; Sarkar and Vafa, 2025). Retrieval compounds the risk: even with date filters, post-cutoff content passes through unreliable metadata (El Lahib et al., 2026). Recent efforts—train-time temporal partitioning (Drinkall et al., 2024; He et al., 2025), benchmark contamination audits (Karger et al., 2025), and memorization detection (Carlini et al., 2021; Li et al., 2024)—mitigate aspects of this problem, yet none provides fine-grained attribution of *which* information constitutes leakage and *how much* it influences predictions.

We address these gaps with two contributions. First, we introduce a model-agnostic, claim-level evaluation framework: a model's prediction rationale is decomposed into atomic claims, each grounded to a temporal provenance. Rather than binary leaked-or-clean labels, we apply Shapley values (Shapley, 1953; Lundberg and Lee, 2017) to quantify each claim's marginal contribution to the prediction, yielding **Shapley-DCLR** (**Shapley**-weighted **D**ecision-**C**ritical **L**eakage **R**ate)—an interpretable metric that captures what fraction of decision-driving reasoning derives from leaked information. Second, we propose **TimeSPEC** (**Time**-**S**upervised **P**rediction with **E**xtracted **C**laims), an inference-time architecture that proactively prevents leakage. TimeSPEC interleaves generation with claim-level supervision; when temporal violations are detected, regeneration produces corrected output using only validated evidence. Unlike prompt-based temporal constraints—which fail because LLMs cannot reliably self-assess the temporal validity of their own knowledge—TimeSPEC enforces validity through programmatic supervision and a closed-world aggregator that admits only verified claims.

We evaluate on 2,769 binary forecasting instances across three LLMs and three prediction domains, applying Shapley-DCLR to every baseline. A 2×2 ablation over (Search, Supervision) reveals that standard prompting and unfiltered retrieval both exhibit substantial leakage, and establishes TimeSPEC's two mechanisms as jointly necessary: neither alone matches the joint objective

of low leakage and retained accuracy. A complementary three-task probe—Stock ranking, NBA Salary, and Legal—shows that the accuracy cost of strict temporal enforcement tracks each task's intrinsic reliance on post-cutoff information, serving as a per-task diagnostic of how much unfiltered accuracy was contamination.

2 Temporal Leakage Evaluation Framework

We introduce a framework for auditing temporal leakage in LLM rationales. The framework is *model-agnostic*, consuming only the rationale and the reference date t_{ref} so that it applies to any LLM without access to model internals. It is *fine-grained*: every atomic claim in the rationale receives an independent leakage decision, in contrast to instance-level contamination metrics that flag whole items. And it is *interpretable*: each leaked claim carries a Shapley-derived weight that quantifies its contribution to the prediction, exposing *which* statements move the decision and *how much*.

2.1 Problem Setup and Temporal Leakage

Consider a prediction task in which an LLM $\mathcal{M}$ forecasts the outcome of an event E resolved at time t_E, conditioned on a reference time $t_{\text{ref}} < t_E$. Backtesting requires that $\mathcal{M}$ reason only from information available before t_{ref}. Let $\mathcal{K}(t)$ denote the cumulative public knowledge at time t, satisfying monotonicity: $t_1 \leq t_2 \implies \mathcal{K}(t_1) \subseteq \mathcal{K}(t_2)$. The model produces a rationale R and a prediction $\hat{y}$. We ask whether R derives solely from $\mathcal{K}(t_{\text{ref}})$ or incorporates information from $\mathcal{K}(t) \setminus \mathcal{K}(t_{\text{ref}})$ for some $t > t_{\text{ref}}$.

For fine-grained analysis, we decompose the rationale into atomic claims $R = \{c_1, \ldots, c_n\}$, each a single verifiable assertion. The *temporal grounding function*

$$\tau(c) = \inf\big\{t : c \in \mathcal{K}(t) \text{ or } c \text{ derivable from } \mathcal{K}(t)\big\} \tag{1}$$

returns the earliest time at which c is publicly knowable, with $\tau(c) = -\infty$ for timeless truths. A claim exhibits *temporal leakage* iff $\tau(c) > t_{\text{ref}}$, recorded by the indicator $\ell(c; t_{\text{ref}}) = \mathbf{1}[\tau(c) > t_{\text{ref}}]$.

2.2 Shapley-Value Decision Weighting

Per-claim leakage is necessary but not sufficient: a leaked claim that does not move the prediction is a less serious failure than one that does. We therefore weight each claim by its contribution to the prediction using Shapley values.

Shapley-value attribution. Let $N = \{1, \ldots, n\}$ index the claims of a rationale and let $v : 2^N \to \mathbb{R}$ be a characteristic function with $v(S)$ equal to the normalized above-baseline prediction score when only claims in subset S are visible to $\mathcal{M}$. The *Shapley value* of claim c_i is

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\,(n-|S|-1)!}{n!} \big[v(S \cup \{i\}) - v(S)\big], \tag{2}$$

the expected marginal contribution of c_i across all orderings in which claims could enter the rationale. The Shapley framework (Shapley, 1953; Lundberg and Lee, 2017) maps cleanly onto our setting: *players* are atomic claims, *coalitions* are claim subsets, and $v(S)$ is the prediction score when the rationale is restricted to S. Shapley values are the unique axiomatic attribution under this characteristic function: they partition the rationale's net contribution into per-claim shares ($\sum_i \phi_i = v(N) - v(\emptyset)$) and assign zero weight to any claim whose marginal contribution to every coalition is zero.

Tractability. Exact computation of $\boldsymbol{\phi}$ requires 2^n coalition evaluations, intractable for typical rationales with 10–20+ claims. Classical Monte Carlo permutation sampling is unbiased but converges only at $\mathrm{Var}(\hat{\phi}_i) = O(1/m)$ without a high-probability precision guarantee. We adopt Leverage SHAP (Musco and Witter, 2025), which reframes Shapley estimation as leverage-weighted regression and admits a provably tight sample complexity:

$$m = O(C \cdot n \cdot \log n) \tag{3}$$

coalitions suffice for a $(1+\epsilon)$-approximation to $\boldsymbol{\phi}$ with high probability, where the constant C controls precision. The exponential budget of exact computation is thus reduced to near-linear in n while the axiomatic properties of Eq. 2 are preserved.

2.3 Leakage Evaluation Pipeline

Given a rationale R and reference date t_{ref}, the framework runs the four-phase pipeline illustrated in Figure 1; Phases 2 and 3 execute in parallel.

Phase 1: Claim extraction and categorization. The extractor decomposes R into atomic claims $\{(c_i, \kappa_i)\}_{i=1}^n$, where each c_i is a single verifiable assertion satisfying atomicity, self-containment,

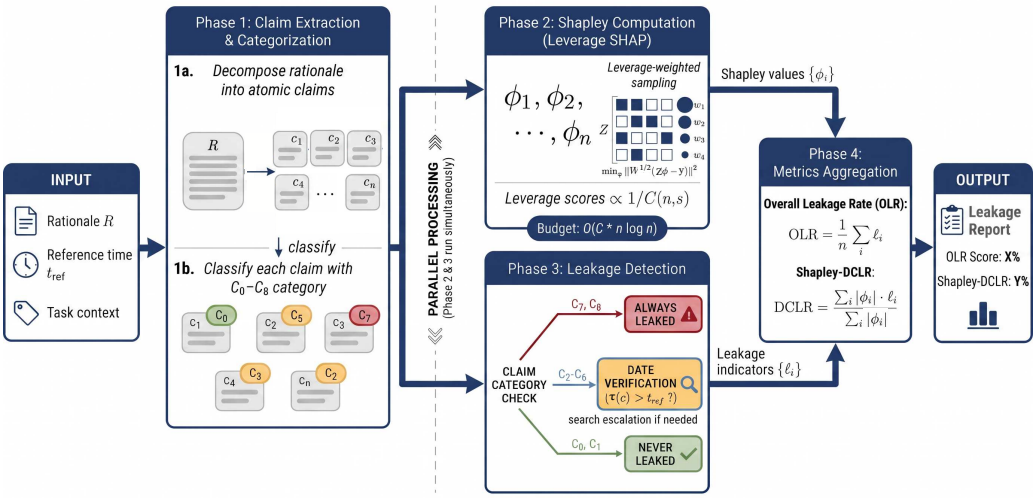

Figure 1: **The Shapley-DCLR evaluation pipeline: a model-agnostic, claim-level framework for detecting and quantifying temporal leakage.** Given any LLM's rationale R and a reference date t_{ref}, **Phase 1** decomposes R into atomic claims and categorizes each. **Phases 2 and 3** run in parallel. Phase 2 estimates each claim's Shapley contribution ϕ_i to the prediction via Leverage SHAP at $O(C \cdot n \log n)$ cost. Phase 3 determines leakage indicators $\{\ell_i\}$ via category-based routing, where deterministic categories bypass external search, reducing verification cost. **Phase 4** fuses $\{\phi_i\}$ and $\{\ell_i\}$ into Shapley-DCLR (Eq. 4).

factuality, and temporal grounding, and $\kappa_i \in \{\text{C0}, \ldots, \text{C8}\}$ is its category under a nine-class taxonomy. The taxonomy partitions claims into three operational tiers—*deterministically safe* (C0–C1), *requires date verification* (C2–C6), and *deterministically leaked* (C7–C8)—enabling the category-based routing in Phase 3. Full specification and worked examples are in Appendix B.

Phase 2: Shapley computation. Phase 2 instantiates the characteristic function v of Section 2.2 by re-running $\mathcal{M}$ on coalitions sampled via Leverage SHAP (Eq. 3), producing per-claim Shapley values $\{\phi_i\}$ at near-linear cost.

Phase 3: Leakage detection. Category-based routing resolves each claim's leakage indicator ℓ_i. Claims whose leakage status is structurally determined are resolved without search. For the remaining claims, the pipeline queries external sources to establish $\tau(c)$ and evaluates $\tau(c) > t_{\text{ref}}$. The authority is the externally-verified publication date of multiple retrieved sources, which is more reliable than model-declared or in-claim dates that can be hallucinated or ambiguous. Vague temporal references are resolved conservatively (e.g., "2023" maps to December 31, 2023) to avoid false negatives from under-specified timestamps.

Phase 4: Metrics aggregation. Phase 4 fuses the Shapley values $\{\phi_i\}$ from Phase 2 with the leakage indicators $\{\ell_i\}$ from Phase 3 into the *Shapley-weighted Decision-Critical Leakage Rate*:

$$\text{Shapley-DCLR} = \frac{\sum_{i=1}^{n} |\phi_i| \cdot \ell(c_i)}{\sum_{i=1}^{n} |\phi_i|}, \qquad (4)$$

which admits two complementary readings. As *influence-weighted leakage*, it measures whether the claims that actually drive the prediction are contaminated—high Shapley-DCLR means decision-critical claims are leaked while a low value indicates peripheral leakage. As *weighted information composition*, it reports the fraction of decision-relevant evidence that derives from post-cutoff sources, enabling both instance-level diagnosis and aggregate quality assessment.

3 TimeSPEC: Inference-Time Forecasting Architecture

TimeSPEC is an inference-time forecasting architecture combining two complementary mechanisms: temporally-filtered retrieval that supplies date-filtered pre-cutoff evidence, and claim-level supervision that audits every generated claim for temporal violations. When violations are detected, a regeneration loop re-queries diverse sources to replace leaked content. A closed-world aggregator

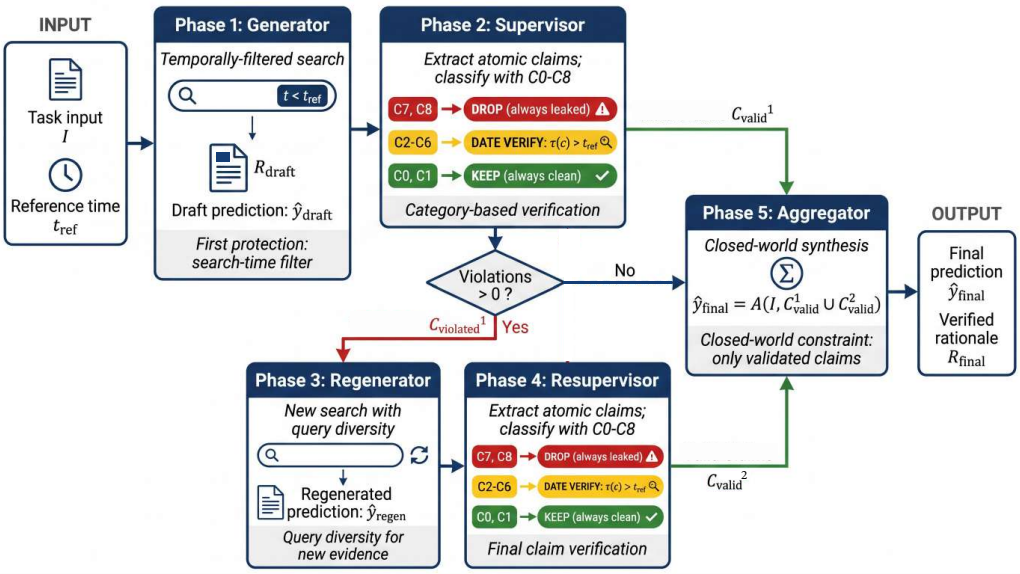

Figure 2: **The TimeSPEC architecture: an inference-time pipeline that interleaves temporally-filtered retrieval with claim-level supervision under a closed-world guarantee.** The **Generator** issues date-filtered web queries ($t < t_{\text{ref}}$) and produces a draft prediction with rationale. The **Supervisor** extracts and categorizes claims, partitioning them into validated ($\mathcal{C}_{\text{valid}}^{(1)}$) and violated ($\mathcal{C}_{\text{violated}}^{(1)}$) sets. If violations exist, the **Regenerator** re-queries with diversified searches and the **Resupervisor** audits the new rationale, yielding $\mathcal{C}_{\text{valid}}^{(2)}$; if none, the validated claims route directly to the **Aggregator**. The Aggregator synthesizes $\hat{y}_{\text{final}} = \mathcal{A}(\mathcal{I}, \mathcal{C}_{\text{valid}}^{(1)} \cup \mathcal{C}_{\text{valid}}^{(2)})$ under a closed-world constraint: only the task input and validated claims enter the final synthesis.

then synthesizes the final prediction from only the task input and validated claims, structurally excluding unaudited parametric content. The five phases are shown in Figure 2.

Phase 1: Generator. Given task input $\mathcal{I}$ and reference date t_{ref}, the Generator issues temporally-filtered web queries with date constraints ($t < t_{\text{ref}}$) and produces a draft prediction $\hat{y}_{\text{draft}}$ with rationale R_{draft}. API-level date filtering provides the first protection layer but is insufficient on its own: date-restricted retrieval still admits post-cutoff content through unreliable metadata (El Lahib et al., 2026), and no retrieval constraint can block the LLM's parametric memory from re-introducing post-cutoff facts at generation time.

Phase 2: Supervisor. To address the limitations of API-level filtering, the Supervisor extracts atomic claims from R_{draft}, categorizes them, and determines each claim's leakage status via category-based routing. Claims are partitioned into a validated set $\mathcal{C}_{\text{valid}}^{(1)}$ and a violated set $\mathcal{C}_{\text{violated}}^{(1)}$. If no violations are found, $\mathcal{C}_{\text{valid}}^{(1)}$ routes directly to Phase 5; otherwise regeneration is triggered.

Phase 3: Regenerator. Activated when violations exist, the Regenerator inherits the validated claims $\mathcal{C}_{\text{valid}}^{(1)}$ and the prior search queries, then issues new queries constrained to be diverse from those already attempted, encouraging retrieval to unexplored pre-cutoff sources. The phase produces an updated prediction $\hat{y}_{\text{regen}}$ with rationale R_{regen}.

Phase 4: Resupervisor. The Resupervisor applies the same claim-level verification as Phase 2 to R_{regen}, producing $\mathcal{C}_{\text{valid}}^{(2)}$. Any residual violations are excluded from aggregation and logged for transparency.

Phase 5: Aggregator. The Aggregator synthesizes the final prediction from the task input and the union of all validated claims:

$$\hat{y}_{\text{final}} = \mathcal{A}\big(\mathcal{I},\ \mathcal{C}_{\text{valid}}^{(1)} \cup \mathcal{C}_{\text{valid}}^{(2)}\big), \tag{5}$$

under a *closed-world* constraint: only $\mathcal{I}$ and the validated claims enter the synthesis prompt, so no unaudited parametric content or unverified retrieved information can influence the final answer. The output is both a prediction $\hat{y}_{\text{final}}$ and a verified rationale R_{final} grounded entirely in dated, categorized evidence.

4 Experiments

The preceding sections introduced two contributions: a claim-level evaluation framework centered

Table 1: Treatment-group results across the (Search, Supervision) ablation on three LLMs. DCLR: Shapley-DCLR (%, ↓, lower is less leakage); Acc: 1−Brier (%, ↑). Temporal RAG achieves the highest accuracy but also the highest leakage on every model; TimeSPEC recovers most of the accuracy gain while suppressing DCLR to the supervised floor. Bold marks the best value per column.

			Qwen3.5-35B		Kimi-K2.5		Claude Sonnet 4	
Method	Search	Sup.	DCLR (%)↓	Acc (%)↑	DCLR (%)↓	Acc (%)↑	DCLR (%)↓	Acc (%)↑
Temporal Hint	–	–	5.2	80.5	2.7	86.8	1.0	83.9
Self-Correction	–	✓	**1.6**	79.9	1.7	86.8	**0.8**	84.0
Temporal RAG	✓	–	13.1	**85.4**	8.9	**87.8**	4.7	**84.2**
TimeSPEC	✓	✓	1.8	83.9	**1.0**	87.4	**0.8**	83.9

Table 2: Control-group Shapley-DCLR (%, ↓) on three LLMs. Control questions resolve after every model's knowledge cutoff, so parametric leakage is impossible by construction. Temporal Hint and Self-Correction confirm the near-zero empirical floor.

Method	Qwen3.5	Kimi-K2.5	Claude
Temporal Hint	0.6	0.5	0.2
Self-Correction	0.5	0.4	0.2
Temporal RAG	16.7	17.0	18.5
TimeSPEC	3.7	2.7	4.2

on Shapley-DCLR (Section 2), and TimeSPEC, which combines temporally-filtered retrieval with claim-level supervision (Section 3). We now validate both empirically. An ablation over (Search, Supervision) on 2,769 instances across three LLMs tests whether the two mechanisms are jointly necessary and, via a Treatment/Control split, validates the metric (Section 4.1). A complementary three-task probe—Stock ranking, NBA Salary, and Legal—shows that the performance effect of strict temporal enforcement tracks each task's intrinsic reliance on post-cutoff information, serving as a per-task diagnostic of how much unfiltered performance was contamination (Section 4.2).

Shared setup. Both experiments use four baselines occupying the cells of a (Search, Supervision) grid. **Temporal Hint** (−Search, −Supervision) issues only a prompt-level date constraint. **Self-Correction** (−Search, +Supervision) adds self-review. **Temporal RAG** (+Search, −Supervision) grants date-filtered retrieval but no claim-level verification. **TimeSPEC** (+Search, +Supervision) is the full pipeline of Section 3. Leakage is reported as Shapley-DCLR (Eq. 4) computed via Leverage SHAP; accuracy metrics are experiment-specific. Full implementation details are in Appendix C; code is available at `https://anonymous.4open.science/r/TimeSPEC-6E24`.

4.1 Large-Scale Ablation

Data and models. The evaluation comprises 2,769 binary forecasting questions partitioned into a Treatment group (1,942 questions from Autocast (Zou et al., 2022), all resolved before every model's knowledge cutoff) and a Control group (827 questions from ForecastBench (Karger et al., 2025), all resolving after every model's cutoff, so parametric leakage is ruled out by construction). Three LLMs span distinct vendors and knowledge cutoffs: Qwen3.5-35B (Qwen Team, 2026), Kimi-K2.5 (Kimi Team and Moonshot AI, 2026), and Claude Sonnet 4 (Anthropic, 2025). Accuracy is 1−Brier, where $\text{Brier} = (\hat{p} - y)^2$, $\hat{p} \in [0, 1]$ is the predicted probability and $y \in \{0, 1\}$ is the outcome; higher is better. Dataset details appear in Appendix A.

Shapley-DCLR is a valid and responsive metric. The Control group establishes the empirical floor: Temporal Hint and Self-Correction produce near-zero DCLR on every model (0.6%/0.5% Qwen3.5, 0.5%/0.4% Kimi, 0.2%/0.2% Claude), confirming that the metric does not flag leakage-free predictions as contaminated. The small residual reflects occasional hallucinated claims that happen to carry post-cutoff dates (e.g., fabricating a future policy announcement), not genuine leakage. Comparing Treatment to Control at Temporal Hint reveals the metric's sensitivity: DCLR rises 8.7× on Qwen3.5 (5.2% vs. 0.6%), 5.4× on Kimi, and 5.0× on Claude once parametric leakage becomes possible. Together, the near-zero Control floor and the several-fold Treatment elevation validate Shapley-DCLR as both precise (few false positives) and sensitive to genuine contamination.

Retrieval improves accuracy but amplifies leakage. Even without search, Temporal Hint already exhibits non-trivial parametric leakage (Treatment DCLR 5.2%/2.7%/1.0%), confirming that mod-

els encode post-cutoff knowledge in their parameters. Adding date-filtered retrieval (Temporal RAG) improves accuracy on every model ($+4.9/+1.0/+0.3$ pts) but amplifies DCLR by 2.5–4.7× (5.2% → 13.1% Qwen3.5, 2.7% → 8.9% Kimi, 1.0% → 4.7% Claude): contamination scales faster than genuine signal. The Control group isolates the retrieval channel directly: with parametric leakage ruled out, Temporal RAG still produces DCLR of 16.7%/17.0%/18.5%, so every leaked claim must originate from retrieved documents. A hard date filter at the search API is therefore insufficient—post-cutoff knowledge propagates through unreliable publication-date metadata and post-publication page edits (El Lahib et al., 2026). Retrieval adds genuine predictive signal, but without claim-level verification it simultaneously introduces a contamination channel that exceeds the parametric baseline.

Both mechanisms in TimeSPEC are jointly necessary. Two pairwise comparisons test whether each mechanism remains non-redundant in the presence of the other. Adding search to Self-Correction (Self-Correction → TimeSPEC) recovers accuracy by up to +4.0 pts on Qwen3.5 (79.9% → 83.9%) while DCLR stays at the supervised floor (1.6% → 1.8%): the supervisor catches the contamination retrieval introduces, so the accuracy gain is decoupled from a leakage gain. Adding supervision to Temporal RAG (Temporal RAG → TimeSPEC) cuts DCLR by 83–89% on every model (13.1% → 1.8%, 8.9% → 1.0%, 4.7% → 0.8%) at bounded accuracy cost (-1.5, -0.4, -0.3 pts). Neither mechanism alone matches TimeSPEC: Self-Correction lacks the evidence retrieval supplies, and Temporal RAG lacks the filter that strips retrieval-borne contamination. The pattern holds on all three models, confirming that the joint-necessity conclusion is not specific to one vendor's training data or alignment recipe.

4.2 TimeSPEC under Varying Leakage Sensitivity

The large-scale ablation validates TimeSPEC's two mechanisms as jointly necessary on average. A natural follow-up is whether the framework honestly reflects a model's true prediction ability on individual tasks—particularly those where temporally valid information alone is insufficient for accurate prediction. On such tasks, models may silently draw on memorized or retrieved post-cutoff knowledge, inflating reported metrics and undermining backtesting integrity. We probe this by constructing three forecasting tasks that deliberately span the leakage-sensitivity spectrum: Stock ranking (high), NBA Salary (intermediate), and Legal (low). For each task, we measure how leakage suppression affects both DCLR and task-specific performance, and analyze what the resulting changes reveal about each task's reliance on post-cutoff knowledge.

Task design. We select three forecasting tasks whose reliance on post-cutoff information varies by construction. *Stock return ranking* is the most leakage-sensitive: market returns are highly volatile and difficult to predict from pre-cutoff fundamentals alone, so models with access to post-cutoff knowledge gain a disproportionate advantage. *NBA Salary prediction* is intermediate: contract values depend on recent player performance and market conditions that may fall within days or weeks of the cutoff, making up-to-date knowledge informative but not as decisive as in stock markets. *Legal case outcome prediction* is least susceptible: judicial outcomes rely primarily on precedent, statutory reasoning, and procedural history available well before the decision, and similar prior cases provide strong valid reference points.

Data, model, and metrics. *Stock*: n=100 ranking instances from the COVID-19 market period (December 2019–June 2020), each containing 5 S&P 500 stocks from a single sector; t_{ref} is set to December 1, 2019 (period start). Performance is $(\rho + 1)/2$ (↑), where ρ is Spearman rank correlation (Spearman, 1904). *NBA Salary*: n=137 notable free-agent contracts spanning 2019–2025; the task is to predict each player's annual average value (AAV) in USD, with t_{ref} set before each contract announcement. Performance is $1 - |\hat{y} - y|/|y|$ (↑). *Legal*: n=100 U.S. Supreme Court cases from recent terms; the task is binary classification of whether the petitioner prevails, with t_{ref} set between oral argument and decision date. Performance is $1 - (\hat{p} - y)^2$ (↑), where $\hat{p} \in [0, 1]$ is the predicted probability and $y \in \{0, 1\}$ the outcome. All baselines run with Qwen3.5-35B. Details are in Appendix A.2.

Stock: performance honestly degrades when leakage suppressed COVID-era returns are driven by events no December-2019 forecaster could anticipate; any model that ranks these stocks well is likely drawing on post-cutoff knowledge.

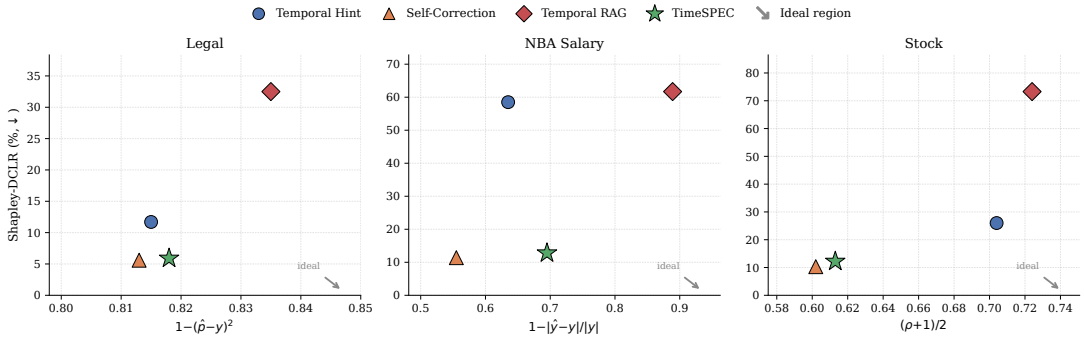

Figure 3: Task-level performance vs. Shapley-DCLR across the four baselines on Qwen3.5-35B. Each panel plots task-specific performance (↑) against Shapley-DCLR (%, ↓); lower-right is the ideal operating region. Performance metrics are $1-(\hat{p}-y)^2$ for Legal, $1-|\hat{y}-y|/|y|$ for NBA Salary, and $(\rho+1)/2$ for Stock, all normalized to $[0, 1]$.

Self-Correction already drops performance sharply relative to Temporal Hint (0.602 vs. 0.704, DCLR 10.3% vs. 26.0%), confirming that the model's memorized COVID knowledge—not pre-cutoff fundamentals—was the primary source of apparent ranking skill. TimeSPEC reaches 0.613 at DCLR 12.2%, a 0.091-point drop from Temporal Hint that measures how much of the unfiltered baseline's ranking ability was parametric contamination. Temporal RAG pushes performance to 0.724 but at DCLR 73.3%—nearly three-quarters of its decision-critical reasoning is leaked—making it the clearest illustration of inflated backtesting. What TimeSPEC reports is what an honest pre-COVID analyst could have produced; the performance drop is not a cost but a correction.

NBA Salary: leakage suppressed, estimation improved. Salary estimation mixes pre-cutoff signal—player statistics, age, cap structure, comparable prior extensions—with post-cutoff targets: the actual signed contracts that an unfiltered model can memorize. Temporal Hint already carries heavy parametric leakage (DCLR 58.5%), as widely-publicized contract values encode directly into model parameters. TimeSPEC strictly dominates Temporal Hint: it reduces DCLR to 12.8% while simultaneously improving estimation accuracy from 0.635 to 0.695, because verified pre-cutoff comparables substitute for leaked contract values. Temporal RAG achieves the highest accuracy (0.889) but at DCLR 61.7%; the 0.194-point gap to TimeSPEC is the contamination premium—most of Temporal RAG's apparent estimation skill rests on leaked post-cutoff contracts, not genuine pre-cutoff reasoning. On a task where pre-cutoff evidence is informative but incomplete, TimeSPEC demonstrates that honest retrieval outperforms leaked memorization.

Legal: leakage suppressed, prediction preserved. Judicial outcomes rely on precedent, statutory text, and prior opinions—an evidentiary base mostly pre-cutoff—so the model has little need to draw on post-cutoff knowledge. All four baselines cluster in a tight performance band (0.813–0.835), and Self-Correction barely differs from Temporal Hint (0.813 vs. 0.815), confirming that parametric leakage contributes negligibly to prediction on this task. Temporal RAG raises DCLR to 32.5% for only a marginal performance gain (0.835 vs. 0.815), showing that even substantial leaked knowledge adds little when valid evidence already suffices. TimeSPEC matches Temporal Hint (0.818 vs. 0.815) at less than half the DCLR (5.9% vs. 11.7%). On a task where temporally valid information suffices, enforcement is effectively free.

The performance effect scales with leakage sensitivity. The three tasks reveal a systematic pattern. The parametric leakage benefit—measured by the Temporal Hint-to-Self-Correction performance drop—grows monotonically with task difficulty: smallest on Legal (−0.002), intermediate on NBA Salary (−0.080), and largest on Stock (−0.102). The harder it is to predict from pre-cutoff evidence alone, the more the model silently relies on memorized post-cutoff knowledge. Comparing Temporal RAG to TimeSPEC isolates the cost of honest supervision over unfiltered retrieval: DCLR drops drastically on every task (32.5%→5.9% Legal, 61.7%→12.8% NBA, 73.3%→12.2% Stock), while the performance gap—0.017 on Legal, 0.194 on NBA, 0.111 on Stock—quantifies how much of Temporal RAG's apparent skill was contami-

nation rather than genuine forecasting. The gap is negligible on Legal, where leaked knowledge adds little, and largest on Stock, where nearly all of Temporal RAG's ranking ability traces to post-cutoff information. The performance change under enforcement is itself a per-task contamination diagnostic: it tells the practitioner how much of the reported performance was genuine.

5 Related Work

Data Contamination and Temporal Knowledge in LLMs. Data contamination—the overlap between training corpora and evaluation benchmarks—inflates performance and undermines reliability (Brown et al., 2020; Carlini et al., 2021; Li et al., 2024). A related but distinct phenomenon is *temporal* contamination: LLMs exhibit temporal blind spots (Wallat et al., 2024), limited awareness of their own knowledge boundaries (Pęzik et al., 2025; Zhu et al., 2025), and degraded performance on temporally shifted inputs (Agarwal and Nenkova, 2022; Dhingra et al., 2022). Recent work formalizes lookahead bias as a structural property of pretrained representations (Sarkar and Vafa, 2025) and proposes contamination-free evaluation through dynamic question generation (Karger et al., 2025; Anonymous, 2025). Our framework addresses a complementary gap: identifying *which* claims constitute leakage and quantifying *how much* each influences the prediction via Shapley attribution.

LLM Backtesting and Temporal Mitigation. Backtesting LLMs on historical events risks conflating hindsight bias with genuine predictive signal (Bailey et al., 2014; Tetlock et al., 2014), and studies in finance confirm that apparent forecasting gains often reflect a "profit mirage" driven by temporal leakage (Li et al., 2025; Yu et al., 2025; Chen et al., 2024). Training-time approaches address this at the model level: Time Machine GPT (Drinkall et al., 2024), ChronoGPT (He et al., 2025), and DATEDGPT (Yan et al., 2026) train on temporally partitioned corpora to eliminate leakage by construction, but require retraining for each cutoff and are limited to models the practitioner controls. At inference time, retrieval-augmented generation (Lewis et al., 2020) is widely adopted but insufficient: date-filtered retrieval can amplify leakage by 2.5–4.7× through unreliable metadata (El Lahib et al., 2026). Our framework is complementary: it operates on any LLM without retraining, combining claim-level verification with temporally-filtered retrieval to suppress both parametric and retrieval-borne contamination.

Shapley Attribution and Interpretable Evaluation. Shapley values (Shapley, 1953) provide axiomatically justified attribution. SHAP (Lundberg and Lee, 2017) and its variants—KernelSHAP (Covert and Lee, 2021) via weighted regression, Leverage SHAP (Musco and Witter, 2025) via leverage-score sampling with provable $(1 + \epsilon)$ guarantees—make estimation practical; we adopt Leverage SHAP for claim-level attribution at scale. On the evaluation side, rationale faithfulness (DeYoung et al., 2020), atomic-fact precision (Min et al., 2023), search-augmented factuality evaluation (Wei et al., 2024), and claim-level entailment verification (Kamoi et al., 2023; Thorne et al., 2018) provide complementary lenses on output quality. Our framework unifies attribution and verification with a temporal dimension: claims are decomposed, temporally classified, verified through date-aware search, and weighted by Shapley-derived decision impact.

6 Conclusion

We presented two contributions for reliable LLM backtesting. First, a model-agnostic evaluation framework decomposes prediction rationales into atomic claims, classifies their temporal provenance, and applies Shapley attribution to quantify each leaked claim's influence on the final decision, yielding Shapley-DCLR as an interpretable contamination metric. Second, TimeSPEC prevents leakage at inference time by coupling temporally-filtered retrieval with claim-level supervision under a closed-world constraint. A large-scale ablation across three LLMs and 2,769 forecasting instances confirms that retrieval and supervision are jointly necessary, and a three-task probe demonstrates that the performance cost of temporal enforcement scales with each task's dependence on post-cutoff information—diagnosing how much of the unfiltered baseline's skill was contamination.

Limitations

TimeSPEC operates at inference time, so leaked information persists in parametric memory and is *blocked* rather than *removed*; temporally partitioned pretraining or reinforcement learning that teaches temporal discipline may address leakage at

the source. The evaluation framework incurs per-instance cost from Shapley coalition evaluations and search-based verification; Leverage SHAP reduces the Shapley budget to $O(n \log n)$ but the overhead remains non-trivial. The TimeSPEC generation pipeline adds further cost through multi-phase supervision and regeneration. Finally, while our experiments span binary forecasting, numerical estimation, and ranking across diverse domains, extending the framework to broader question types—particularly open-ended generation—remains future work.

References

Oshin Agarwal and Ani Nenkova. 2022. Temporal effects on pre-trained models for language processing tasks. *Transactions of the Association for Computational Linguistics*, 10:904–921.

Anonymous. 2025. Forecasting with LLMs: A dataset for rapid backtesting without temporal contamination. In *ICLR 2026 Conference Submission*. Under review.

Anthropic. 2025. System card: Claude Opus 4 & Claude Sonnet 4. *Anthropic Technical Report*.

David H Bailey, Jonathan M Borwein, Marcos López de Prado, and Qiji Jim Zhu. 2014. Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance. *Notices of the American Mathematical Society*, 61(5):458–471.

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.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Shuaiyu Chen, T Clifton Green, Huseyin Gulen, and Dexin Zhou. 2024. What does chatgpt make of historical stock returns? extrapolation and miscalibration in llm stock return forecasts. *arXiv preprint arXiv:2409.11540*.

Ian Covert and Su-In Lee. 2021. Improving KernelSHAP: Practical Shapley value estimation using linear regression. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3457–3465.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2020. Eraser: A benchmark to evaluate rationalized nlp models. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4443–4458.

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.

Felix Drinkall, Eghbal Rahimikia, Janet B Pierrehumbert, and Stefan Zohren. 2024. Time machine GPT. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3281–3292.

Ali El Lahib, Ying-Jieh Xia, Zehan Li, Yuxuan Wang, and Xinyu Pi. 2026. Temporal leakage in search-engine date-filtered web retrieval: A retrospective forecasting case study. *arXiv preprint arXiv:2602.00758*.

Songrun He, Linying Lv, Asaf Manela, and Jimmy Wu. 2025. Chronologically consistent large language models. *arXiv preprint arXiv:2502.21206*.

Ryo Kamoi, Tanya Goyal, Juan Diego Rodriguez, and Greg Durrett. 2023. WiCE: Real-world entailment for claims in Wikipedia. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7561–7583.

Ezra Karger, Houtan Bastani, Chen Yueh-Han, Zachary Jacobs, Danny Halawi, Fred Zhang, and Philip E Tetlock. 2025. Forecastbench: A dynamic benchmark of ai forecasting capabilities. In *The Thirteenth International Conference on Learning Representations*.

Kimi Team and Moonshot AI. 2026. Kimi k2.5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

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.

Xiangyu Li, Yawen Zeng, Xiaofen Xing, Jin Xu, and Xiangmin Xu. 2025. Profit mirage: Revisiting information leakage in llm-based financial agents. *arXiv preprint arXiv:2510.07920*.

Yucheng Li, Yunhao Guo, Frank Guerin, and Chenghua Lin. 2024. An open-source data contamination report for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 528–541.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30.

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. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.

Christopher Musco and R. Teal Witter. 2025. Provably accurate shapley value estimation via leverage score sampling. In *The Thirteenth International Conference on Learning Representations*.

Piotr Pęzik, Konrad Kaczyński, Maria Szymańska, Filip Żarnecki, Zuzanna Deckert, Jakub Kwiatkowski, and Wojciech Janowski. 2025. LLMLagBench: Identifying temporal training boundaries in large language models. *arXiv preprint arXiv:2511.12116*.

Qwen Team. 2026. Qwen3.5: Towards native multimodal agents.

Suproteem K Sarkar and Keyon Vafa. 2025. Lookahead bias in pretrained language models. In *ICML 2025 Workshop on Reliable and Responsible Foundation Models*.

Lloyd S Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317.

Charles Spearman. 1904. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101.

Philip E Tetlock, Barbara A Mellers, Nick Rohrbaugh, and Eva Chen. 2014. Forecasting tournaments: Tools for increasing transparency and improving the quality of debate. *Current Directions in Psychological Science*, 23(4):290–295.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The fact extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 1–9.

Jonas Wallat, Adam Jatowt, and Avishek Anand. 2024. Temporal blind spots in large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 683–692.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. 2024. Long-form factuality in large language models. *Preprint*, arXiv:2403.18802.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*.

Yutong Yan, Raphael Tang, Zhenyu Gao, Wenxi Jiang, and Yao Lu. 2026. DATEDGPT: Preventing lookahead bias in large language models with time-aware pretraining. *arXiv preprint arXiv:2603.11838*.

Haofei Yu, Fenghai Li, and Jiaxuan You. 2025. Livetradebench: Seeking real-world alpha with large language models. *arXiv preprint arXiv:2511.03628*.

Chenghao Zhu, Nuo Chen, Yufei Gao, Yunyi Zhang, Prayag Tiwari, and Benyou Wang. 2025. Is your LLM outdated? a deep look at temporal generalization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7433–7457.

Andy Zou, Tristan Xiao, Ryan Jia, Joe Kwon, Mantas Mazeika, Richard Li, Dawn Song, Jacob Steinhardt, Owain Evans, and Dan Hendrycks. 2022. Forecasting future world events with neural networks. In *Advances in Neural Information Processing Systems*, volume 35.

A Dataset Details

This section describes the datasets used in both experiments. Section A.1 covers the large-scale ablation and Section A.2 covers the task-sensitivity probe.

A.1 Large-Scale Ablation

The ablation uses 2,769 binary forecasting questions partitioned into Treatment and Control groups by temporal accessibility relative to model knowledge cutoffs.

Treatment group. The Treatment group comprises 1,942 binary questions drawn from Autocast (Zou et al., 2022), a forecasting benchmark of 6,707 questions collected from three public forecasting tournaments (Metaculus, Good Judgment Open, and CSET Foretell). Questions span geopolitics, economics, science, and public health, each with a ground-truth resolution and a known resolution date. We retain the binary-outcome subset whose resolution dates precede the knowledge cutoff of every model tested (Qwen3.5-35B, Kimi-K2.5, Claude Sonnet 4), so parametric leakage from training data is possible.

Control group. The Control group comprises 827 binary questions drawn from ForecastBench (Karger et al., 2025), a dynamic benchmark that ingests questions nightly from nine sources—four forecasting platforms (Manifold, Metaculus, Polymarket, RAND Forecasting Initiative) and five real-world datasets (ACLED, DBnomics, FRED, Wikipedia, Yahoo Finance)—and samples fresh question sets every two weeks. By construction, all Control questions resolve after every model's knowledge cutoff, so parametric leakage is impossible. This group serves three purposes: it validates that Shapley-DCLR does not produce false positives on leakage-free instances, isolates retrieval as a leakage channel by removing the parametric confound, and establishes a lower bound on achievable DCLR.

Table 3 summarizes the two groups.

A.2 Task-Sensitivity Probe

The probe comprises three forecasting tasks that span the leakage-sensitivity spectrum.

Stock return ranking. We construct 100 ranking instances from the COVID-19 market period (December 2019–June 2020). Each instance contains 5 S&P 500 stocks from a single GICS sector; the task is to rank them by realized six-month return. The reference date is fixed at t_{ref} = December 1, 2019, which precedes public awareness of COVID-19 (China alerted the WHO on December 31, 2019). This design makes the task deliberately leakage-sensitive: accurate ranking requires knowledge of pandemic-driven market dynamics that were unknowable at t_{ref}. Performance is $(\rho + 1)/2$, where ρ is Spearman rank correlation (Spearman, 1904), normalized to $[0, 1]$. Stock price data are from Yahoo Finance.[1]

NBA salary prediction. We compile 137 notable NBA free-agent contracts spanning 2019–2025 from public transaction records.[2] The task is to predict each player's annual average value (AAV) in USD. Input features include the player's previous team, position, and career statistics. For each instance, t_{ref} is set before the contract announcement, ensuring that the signed contract details—and contemporaneous peer deals—are post-cutoff. This task has intermediate leakage sensitivity: pre-cutoff evidence (player statistics, age, prior extensions, salary cap projections) is informative, but the exact contract values that an unfiltered model can memorize are post-cutoff. Performance is $1 - |\hat{y} - y|/|y|$, higher is better.

Legal case outcome prediction. We curate 100 U.S. Supreme Court cases from recent terms.[3] The task is binary classification: predict whether the petitioner prevails. Input features include case name, parties, legal question, and factual background. For each case, t_{ref} is set between oral argument and the decision date, so the ruling itself is post-cutoff while all briefing, argument transcripts, and prior precedent are available. This task is the least leakage-sensitive: judicial outcomes rely primarily on precedent established years or decades earlier, statutory text fixed at enactment, and judicial philosophies observable through prior opinions. Performance is $1 - (\hat{p} - y)^2$, where $\hat{p} \in [0, 1]$ is the predicted probability and $y \in \{0, 1\}$ the outcome, higher is better.

Table 4 summarizes the three tasks.

[1] https://finance.yahoo.com

[2] Salary data from Spotrac: https://www.spotrac.com/nba

[3] Case data from the Supreme Court Database and Oyez: https://www.oyez.org

Table 3: Large-scale ablation dataset summary. Treatment questions admit parametric leakage; Control questions do not.

Group	Source	N	Format	Leakage Status
Treatment	Autocast (Zou et al., 2022)	1,942	Binary	Possible (pre-cutoff resolution)
Control	ForecastBench (Karger et al., 2025)	827	Binary	Impossible (post-cutoff resolution)
Total		2,769		

Table 4: Task-sensitivity probe dataset summary. Tasks are ordered from highest to lowest leakage sensitivity.

Task	Type	N	Period	Cutoff Design	Metric
Stock	Ranking	100	Dec 2019–Jun 2020	Period start	$(\rho{+}1)/2$ (↑)
Salary	Regression	137	2019–2025	Pre-signing	$1{-}\lvert\hat{y}{-}y\rvert/\lvert y\rvert$ (↑)
Legal	Classification	100	Recent terms	Pre-decision	$1{-}(\hat{p}{-}y)^2$ (↑)

B Claim Taxonomy: Full Specification

Evaluating the leakage indicator $\ell(c; t_{\text{ref}}) = \mathbf{1}[\tau(c) > t_{\text{ref}}]$ for an arbitrary natural-language claim requires establishing the earliest date at which c is publicly knowable, then comparing against t_{ref}. Most claims, however, fall into types whose leakage status is determined by structure alone, bypassing external lookup entirely. We formalize this observation as a nine-class taxonomy $\{\text{C0}, \ldots, \text{C8}\}$ that partitions all admissible factual claims. Table 5 lists all nine categories with examples drawn from our forecasting tasks and the verification rule applied by Phase 3 of the evaluation pipeline.

The taxonomy satisfies two properties that together guarantee a well-defined classification for every claim.

Proposition 1 (Exclusivity) *Let $\mathcal{C} = \{C0, \ldots, C8\}$ be the nine claim categories, and define the strict precedence order*

$$C7 \succ C8 \succ C6 \succ C3 \succ C4 \succ C5 \succ C2 \succ C1 \succ C0. \tag{6}$$

For any factual claim c that satisfies the membership criteria of two or more categories $\kappa_1, \kappa_2 \in \mathcal{C}$, the classifier assigns the unique label $\kappa^* = \arg\max_{\succ}\{\kappa_1, \kappa_2\}$. Thus every claim receives exactly one category.

Justification. The precedence order ranks categories by leakage severity: C7–C8 (deterministically leaked) dominate C2–C6 (date-dependent), which in turn dominate C0–C1 (deterministically safe). Within each tier, higher precedence goes to categories carrying stronger temporal signal—for instance, the target outcome (C7) outranks post-outcome consequences (C8), and publication-sourced claims (C6) outrank generic time-indexed states (C2). When a claim matches multiple categories, the $\arg\max$ under $\succ$ selects the unique highest-severity label, so no ambiguity arises. □

Proposition 2 (Completeness) *For any admissible factual claim c extracted from a prediction rationale, there exists at least one $\kappa \in \mathcal{C}$ whose membership criteria c satisfies. Completeness is guaranteed by C2 (Time-Indexed State), which serves as the residual class: any time-dependent factual assertion not captured by C3–C8 falls into C2 by default.*

Justification. Categories C3–C8 capture specific claim types (quantitative measurements, events, publications, target outcomes, and their consequences), while C0–C1 capture time-invariant facts. Any remaining factual claim with temporal content—a condition, status, or trend indexed to a specific time period—satisfies C2's membership criterion by definition. Because C2 subsumes all temporal assertions not already captured by the specialized categories, no admissible claim is left unclassified. □

Together, Propositions 1 and 2 ensure that the taxonomy induces a partition over all admissible claims. This partition yields three *operational tiers* that directly govern the verification cost within our leakage detection framework. Categories C0 and C1 are *deterministically safe*: their temporal grounding satisfies $\tau(c) \leq t_{\text{ref}}$ by definition, so the pipeline assigns $\ell(c) = 0$ without invoking any external search. Symmetrically, C7 and C8 are *deterministically leaked*: the target event and its downstream effects are post-cutoff by construction, so $\ell(c) = 1$ is assigned immediately. Only claims

Table 5: Full nine-class claim taxonomy. The *Status* column indicates how $\ell(c)$ is determined: deterministic categories (C0–C1, C7–C8) bypass external search entirely, while C2–C6 require date verification via $\tau(c)$. Horizontal rules separate the three operational tiers.

ID	Type	Example Claim	Status	$\tau(c)$ Rule
C0	Timeless Definition	*"The Brier score is $(p-o)^2$ for predicted probability p and binary outcome o."*	Never leaked	$\tau(c) = -\infty$; time-invariant by definition.
C1	Stable Entity Attribute	*"The U.S. Supreme Court is composed of nine justices."*	Usually safe	Deterministic; attribute must be stable through t_{ref}.
C2	Time-Indexed State	*"As of Q3 2019, U.S. crude-oil demand had been declining for three consecutive quarters."*	Verify date	Earliest source date corroborating the stated condition.
C3	Quantitative Measurement	*"Apple's FY2019 revenue was $260.2B."*	Verify date	Date of public release (e.g., 10-K filing date).
C4	Comparative / Relational	*"In 2019, Boeing's debt-to-equity ratio exceeded that of Airbus by >3×."*	Verify date	Latest source date among the comparands.
C5	Discrete Event	*"On Oct. 25, 2019, Microsoft won the $10B JEDI cloud contract."*	Verify date	Event date, or announcement date if unreported.
C6	Publication / Source	*"A Sep. 15, 2019 Reuters article reported Ford's market share at 13.4%."*	Verify date	Publication date of the cited source.
C7	Target Outcome	*"Durant signed a 4-yr, $164M deal with the Nets."* (when salary AAV is the prediction target)	Always leaked	$\tau(c) = t_E > t_{\text{ref}}$ by task definition.
C8	Post-Outcome Consequence	*"The signing pushed the Nets' 2019–20 payroll above the luxury-tax threshold."*	Always leaked	$\tau(c) > t_E > t_{\text{ref}}$; exists only after the target resolves.

in the intermediate categories C2–C6 require date verification—the pipeline retrieves the earliest publicly available source for each claim and compares its publication date against t_{ref} to determine $\ell(c)$. Empirically, the two deterministic tiers account for 30–50% of extracted claims across our experiments, yielding a corresponding reduction in the number of search queries required per instance.

Worked example. We illustrate the taxonomy on a real instance from our large-scale ablation. The forecasting question *"Will Elon Musk be sanctioned for Tesla buyout tweet?"* has reference date t_{ref} = September 1, 2018. Qwen3.5-35B produces a rationale containing seven atomic claims. Three claims describe pre-cutoff events—Musk's August 7 tweet, the subsequent stock-price surge, and the SEC's August 22 investigation announcement—all classified **C0** and assigned $\ell(c) = 0$ deterministically without search. Three further claims reference the SEC settlement: the model declares dates in late August, but date verification corrects these to September 29 (post-cutoff), classifying all three as **C4** with $\ell(c) = 1$. The remaining claim—*"The settlement required Musk to pay a $20M fine, which exceeds the $1M threshold specified in the question"*—directly reveals the prediction target and is classified **C7** deterministically. This claim also qualifies as C3 (quantitative measurement), but the precedence order (C7 $\succ$ C3, Eq. 6) assigns C7 as the unique label.

The resulting Shapley-DCLR is 0.462: nearly half of the decision-driving reasoning is contaminated, with the C7 claim carrying the highest Shapley value ($\phi = 0.232$). Operationally, only the three C4 claims require external search; the four deterministic claims (three C0 and one C7) are resolved by category alone, reducing the verification budget by 57%.

C Method Implementations

This section details the evaluation pipeline (Section C.1) and the four baseline generation methods (Section C.2). Code and configurations are publicly available.[4] Prompt specifications are in Appendix F.

[4] https://anonymous.4open.science/r/TimeSPEC-6E24

C.1 Evaluation Pipeline

The evaluation pipeline runs post-hoc on the rationale produced by any generation method. It takes as input a structured rationale containing atomic evidence claims (each with a `fact`, `source_date`, and `id`) and the instance's reference date t_{ref}, and outputs per-claim leakage decisions and Shapley-weighted decision-critical leakage rates.

Claim Extraction and Taxonomy Classification (Phase 1). We employ Claude Sonnet 4[5] (T=0, `max_tokens`=8000) to decompose each rationale into atomic claims. Each extraction call receives the rationale text, task description, event description, and t_{ref} as context. The extractor outputs structured objects containing claim text, original text span, temporal reference, C0–C8 category label, and category reasoning. The taxonomy classification follows Appendix B and applies the precedence order (Eq. 6) when a claim matches multiple categories.

Shapley Value Computation (Phase 2). Shapley values are computed via Leverage SHAP (Musco and Witter, 2025) following the reference implementation.[6] The coalition sampler uses uniform per-size weights with paired complements and without-replacement sampling. The budget parameter is C=15 with a cap of 1,200 coalitions; for instances with $n \leq 8$ claims, exact enumeration (2^n coalitions) is used instead. Each coalition is evaluated with a closed-world prompt that provides only the claims in the coalition subset; the prediction model matches the original generator (same OpenRouter model ID, T=0). The projected weighted regression enforces the efficiency axiom ($\sum_i \phi_i = v(N) - v(\emptyset)$) by construction. Coalition predictions are cached to avoid redundant model calls when multiple coalitions share the same claim subset. Concurrency is set to 8 parallel coalition evaluations.

Leakage Detection (Phase 3). Leakage detection applies the three-tier routing from Appendix B to minimize verification cost. C7 and C8 claims are marked leaked deterministically; C0 and C1 claims are marked safe deterministically. Only C2–C6 claims undergo date verification. For these, a taxonomy-aware LLM verifier (Gemini 2.0 Flash Lite,[7] T=0) first assesses whether the declared source date is plausible given the claim content. Claims with implausible or missing dates are escalated to search-based verification via Perplexity API[8]: the pipeline retrieves up to 3 results per claim and a second LLM pass extracts the earliest reliable public availability date from the search snippets. Coarse temporal references are normalized conservatively (e.g., "2023" → 2023-12-31, "Q3 2023" → 2023-09-30) to avoid false negatives. The final leakage decision compares the effective source date against t_{ref}.

C.2 Generation Methods

The four methods occupy the cells of a 2×2 (Search, Supervision) grid, as described in Section 4. All methods share a common generation backbone accessed through OpenRouter,[9] with T=0 and `max_tokens`=4096. The pipeline exploits two shared generation phases to minimize redundant API calls: Phase 1A generates rationales without retrieval context (shared by Temporal Hint and Self-Correction), and Phase 1B generates rationales with retrieval context (shared by Temporal RAG and TimeSPEC). Output parsing and schema validation are identical across all methods.

Temporal Hint (−Search, −Supervision). The simplest baseline: the model receives the task input and a prompt-level temporal constraint ("Your knowledge cutoff is `{date}`. Only use information available on or before this date.") and produces a single-pass prediction with rationale. No search tools, no supervision, no regeneration. This tests whether prompt engineering alone can prevent temporal leakage from the model's parametric memory.

Self-Correction (−Search, +Supervision). Self-Correction extends Temporal Hint with an LLM-only claim-level supervisor. After initial generation (shared Phase 1A), the supervisor classifies each evidence claim under the C0–C8 taxonomy and flags leaked claims using the LLM verifier alone (no external search). If leaked claims are detected, the method triggers regeneration: the model receives the validated (non-leaked) claims from all prior cycles and generates a new rationale. The cycle repeats up to `max_regen_cycles`=1 time. When any leakage was detected across

[5] https://openrouter.ai/anthropic/claude-sonnet-4

[6] https://github.com/rtealwitter/leverageshap

[7] https://openrouter.ai/google/gemini-2.0-flash-lite-001

[8] https://www.perplexity.ai

[9] https://openrouter.ai

cycles, a closed-world aggregator synthesizes the final prediction from only the union of validated claims (deduplicated by fact text), ensuring no leaked content persists. The key architectural difference from TimeSPEC is the absence of retrieval: the supervisor operates entirely on the model's own claims without external date verification.

Temporal RAG (+Search, −Supervision). Temporal RAG augments the generation prompt with date-filtered retrieval context but applies no post-hoc claim verification. The retrieval pipeline first generates task-specific search queries via an LLM, then issues them to Perplexity API with a temporal cutoff constraint ($t < t_{\text{ref}}$), retrieves up to 5 results per query, and assembles a context block (budget: 3,000 characters) that is prepended to the generation prompt. The model produces a single-pass prediction conditioned on both the context and the temporal hint. This method tests whether retrieval with API-level date filtering suffices to prevent leakage—our experiments show it does not, as post-cutoff content passes through unreliable metadata (El Lahib et al., 2026).

TimeSPEC (+Search, +Supervision). The full pipeline described in Section 3. TimeSPEC combines the retrieval context of Temporal RAG with the claim-level supervision of Self-Correction, and adds two architectural enhancements. First, the supervisor uses *search-escalated verification*: after the initial LLM-only taxonomy pass, claims with uncertain or borderline dates are escalated to Perplexity search for independent date confirmation, then re-verified against the search snippets by a second LLM pass. This catches cases where the model backdates post-cutoff events (as illustrated in the worked example of Appendix B). Second, a *defense-in-depth evidence scrub* runs after the aggregator: any claim whose source date exceeds t_{ref} or that was flagged by the last supervisor pass is stripped from the final evidence list, guarding against the aggregator hallucinating fresh post-cutoff facts. Regeneration follows the same cycle as Self-Correction (`max_regen_cycles`=1), and the closed-world aggregator is always invoked when any leakage was detected.

Table 6 summarizes the architectural differences across the four methods.

D Leakage Concentration in Decision-Critical Claims

Shapley-DCLR (Eq. 4) quantifies the aggregate fraction of decision weight attributable to leaked claims, but does not reveal whether contamination concentrates among the most influential claims or spreads thinly across many low-impact ones. The distinction matters: a 20% DCLR carried entirely by the single highest-Shapley claim poses a greater integrity risk than the same 20% distributed across peripheral reasoning. We examine this concentration effect by computing Top-K leakage rates across all four baselines and three tasks from the task-sensitivity probe (Section 4.2).

Definition. For each instance, we rank claims by Shapley value magnitude $|\phi_i|$ in descending order and compute the leakage rate among the top-K most influential claims:

$$\text{Top-}K \text{ Leakage} = \frac{1}{K} \sum_{i=1}^{K} \ell\big(c_{(i)}\big), \qquad (7)$$

where $c_{(i)}$ denotes the claim with the i-th highest Shapley magnitude and $\ell(\cdot) \in \{0, 1\}$ is the leakage indicator. Instances with zero leaked claims receive Top-$K = 0$ for all K. We report the mean across all instances per task, alongside Shapley-DCLR for reference. If Top-K exceeds DCLR, leaked claims are over-represented at the top of the influence ranking.

Table 7 reports the results. All runs use Qwen3.5-35B.

Legal: low leakage, no concentration. All four methods produce DCLR below 12%, and Top-K rates remain in the same range. The gap between Top-1 and DCLR is modest even for Temporal RAG (41.0% vs. 32.5%), and the two supervised methods bring Top-1 below 7%. Legal reasoning relies on precedent and statutory text available well before the decision date, so the few leaked claims that appear are not systematically the most influential ones. Supervision reduces both the aggregate and the peak with little differential effect.

NBA Salary: strong concentration in unsupervised methods. Salary prediction exhibits the most pronounced concentration effect. Temporal Hint's Top-1 leakage (70.8%) exceeds its DCLR (58.5%) by 12 percentage points: in over 70% of instances, the single most influential claim is leaked—typically the actual signed contract value

Table 6: Architectural comparison of the four generation methods. ✓ indicates the component is present; – indicates absent.

Method	Prompt	RAG	Supervisor	Search	Regen	Aggregator
Temporal Hint	✓	–	–	–	–	–
Self-Correction	✓	–	LLM-only	–	✓	✓
Temporal RAG	✓	✓	–	–	–	–
TimeSPEC	✓	✓	LLM+Search	✓	✓	✓

Table 7: Leakage concentration among the most decision-critical claims. *DCLR* is the Shapley-weighted aggregate from Section 4.2; *Top-K* is the fraction of the K highest-Shapley claims that are leaked (Eq. 7), averaged across instances. All values in %. Top-1 > DCLR indicates that leakage concentrates at the most influential position. Bold marks the lowest value per column within each task group.

	Legal (n=100)				**NBA Salary** (n=137)				**Stock** (n=100)			
Method	DCLR↓	Top-1↓	Top-3↓	Top-5↓	DCLR↓	Top-1↓	Top-3↓	Top-5↓	DCLR↓	Top-1↓	Top-3↓	Top-5↓
Temporal Hint	11.7	17.0	11.3	10.1	58.5	70.8	50.4	48.2	26.0	28.0	26.7	26.2
Self-Correction	**5.6**	7.0	**5.0**	**4.0**	**11.3**	**11.7**	**9.7**	**9.0**	**10.2**	**7.0**	8.7	8.6
Temporal RAG	32.5	41.0	35.3	30.7	61.7	75.2	59.1	52.0	73.3	80.0	74.3	72.8
TimeSPEC	5.9	**5.0**	5.7	5.8	12.8	14.6	12.5	10.6	12.2	**7.0**	**7.0**	**7.0**

or a contemporaneous comparable deal that the model memorized from training data. Temporal RAG amplifies this further (Top-1 = 75.2%, DCLR = 61.7%), as retrieved post-cutoff contract details surface as high-Shapley evidence. Self-Correction reduces Top-1 to 11.7% and TimeSPEC to 14.6%, both close to their respective DCLRs (11.3%, 12.8%), indicating that claim-level supervision removes leaked claims specifically from the high-impact positions rather than merely suppressing peripheral contamination. The declining Top-1 → Top-5 gradient for Temporal Hint (70.8% → 48.2%) and Temporal RAG (75.2% → 52.0%) confirms that leakage concentrates at the top: diluting across more claims lowers the rate, meaning the very first claim is disproportionately likely to be contaminated.

Stock: retrieval creates extreme top-rank contamination. On the most leakage-sensitive task, Temporal RAG exhibits 80.0% Top-1 leakage—in four out of five instances, the single most influential claim is leaked knowledge of COVID-era market movements. The rate barely declines to 72.8% at Top-5, indicating that contamination saturates the entire upper stratum of the influence ranking, consistent with this method's 73.3% DCLR. Temporal Hint shows a weaker but still present concentration effect (Top-1 = 28.0% vs. DCLR = 26.0%). Both supervised methods sharply reduce Top-1 to 7.0%. TimeSPEC maintains a flat 7.0% across all K values, while Self-Correction shows slightly higher Top-3 and Top-5 (8.7%, 8.6%)—indicating that TimeSPEC's search-escalated verification removes contamination more thoroughly from the full upper stratum, not just the peak position.

Summary. Across all three tasks, unsupervised methods exhibit a consistent pattern: Top-1 leakage exceeds DCLR, confirming that leaked claims are over-represented at the most influential position. The concentration effect is strongest on tasks with high leakage sensitivity (Salary, Stock) and weakest where valid pre-cutoff evidence suffices (Legal). Claim-level supervision—whether LLM-only (Self-Correction) or search-escalated (TimeSPEC)—eliminates this concentration: Top-K rates converge to DCLR, meaning residual leakage no longer preferentially occupies decision-critical positions. The practical implication is that DCLR alone understates the integrity risk of unsupervised predictions: the contaminated fraction disproportionately drives the final answer.

E Case Studies

We present one representative instance from each task to illustrate how temporal leakage manifests in model rationales, how Shapley-DCLR exposes the contaminated reasoning that drives predictions, and how TimeSPEC suppresses it. All four methods run on Qwen3.5-35B.

E.1 Stock Return Ranking: Healthcare Sector

We examine a ranking instance from the Health Care sector: five stocks—Moderna (MRNA), Bio-Techne (TECH), Becton Dickinson (BDX),

Stryker (SYK), and Dexcom (DXCM)—ranked by six-month return from December 1, 2019 to June 2, 2020. The reference date is t_{ref} = December 1, 2019, which precedes any public knowledge of COVID-19 (China alerted the WHO on December 31, 2019). The ground-truth ranking by realized returns is MRNA > DXCM > TECH > SYK > BDX, driven almost entirely by the pandemic: Moderna's mRNA vaccine program propelled MRNA to the top, while medical-device names lagged.

Temporal Hint and Temporal RAG both achieve a perfect ranking ($\rho = 1.0$), and Self-Correction achieves $\rho = 0.95$ (swapping only BDX and SYK). At face value, this suggests strong forecasting ability. Shapley-DCLR tells a different story. Temporal Hint's DCLR is 55.7%: the rationale contains nine extracted claims, five of which are leaked. The most influential leaked claims state that "Moderna's stock price rose from roughly $100 in December 2019 to over $200 by late February 2020" (C8, $\phi = 0.124$), that "on January 31, 2020, Moderna announced the successful isolation of the SARS-CoV-2 virus and the design of its mRNA vaccine candidate" ($\phi = 0.126$), and that "the global outbreak of COVID-19 in early 2020 created unprecedented demand for vaccine development" ($\phi = 0.112$). These claims—Moderna's post-cutoff stock trajectory, the vaccine announcement two months after t_{ref}, and the pandemic itself—encode the very mechanism that produced the ground-truth ranking. The remaining safe claims describe pre-cutoff fundamentals for the non-biotechnology names (Bio-Techne's steady growth, Dexcom's CGM market position, BDX and SYK as lower-volatility device companies), which alone do not predict Moderna's pandemic-era dominance. The 55.7% DCLR quantifies this: over half the decision weight rests on information that did not exist on December 1, 2019, and the "perfect" ranking is a product of temporal contamination rather than analytical skill.

Self-Correction reduces DCLR from 55.7% to 30.5% by removing the three most explicit post-cutoff claims—the COVID-19 outbreak, the Moderna stock trajectory, and the January 2020 vaccine announcement. Two subtler leaked claims survive: one referencing Moderna "trading at approximately $100 per share" with a verified source date in January 2020, and another alluding to a "vaccine development catalyst" sourced from March 2020. Both are phrased as general market observations rather than dated events, evading the LLM-only verifier. The ranking remains near-perfect (0.95), indicating that even partial leakage suffices for accurate ranking when the leaked information encodes the dominant market driver.

Temporal RAG amplifies contamination to its extreme: DCLR = 100%, with all eight extracted claims leaked. The retrieved content includes explicit stock prices from December 2019 through December 2020, pandemic-driven sector commentary, and year-end performance summaries. The highest-Shapley claim ($\phi = 0.193$) states that established companies' "returns are expected to be lower than Moderna's explosive growth driven by COVID-19 vaccine development," directly encoding the pandemic narrative as retrieved evidence. The second-highest claim references Moderna's "major surge in 2020" ($\phi = 0.178$). API-level date filtering failed entirely: post-cutoff articles with unreliable metadata passed through, and the model assembled a rationale constructed entirely from future information. The perfect ranking carries zero predictive content.

TimeSPEC achieves DCLR = 0.0%. Its search-escalated verification blocks all pandemic-era content, and the validated claims are grounded exclusively in pre-cutoff fundamentals: Moderna's post-IPO recovery in 2019, Bio-Techne's fiscal 2019 results released in July 2019, Stryker's February 2019 10-K filing, Dexcom's November 2019 market capitalization, and BDX's 2019 corporate governance activity. The resulting ranking places MRNA first—justified by its strong 2019 recovery trajectory—but ranks DXCM fourth rather than second, because no pre-cutoff evidence predicts the pandemic-driven surge in continuous glucose monitoring demand. The lower Spearman correlation ($\rho = 0.85$, performance = 0.85) reflects what an analyst could legitimately infer from Q3 2019 earnings alone. This is precisely the outcome honest backtesting should produce: without foreknowledge of a black swan event, perfect ranking of pandemic beneficiaries is unjustified, and the "underperformance" is a correct diagnostic that the task's apparent predictability was an artifact of contamination.

E.2 NBA Salary Prediction: Kevin Durant (2019)

Kevin Durant (SF, Golden State Warriors) entered the 2019 free agency and signed a 4-year, $164.3M contract with the Brooklyn Nets on June 30, 2019 (AAV: $41.06M). The reference date is t_{ref} =

June 23, 2019—one week before the signing. This narrow temporal margin makes the case particularly revealing: pre-cutoff information (CBA rules, Durant's service years, market projections) is sufficient for a reasonable estimate, but the exact contract terms were announced only seven days after t_{ref}.

Temporal Hint, Self-Correction, and Temporal RAG all predict within 0.2% of the ground truth ($41.0M, $41.0M, and $41.08M respectively), while TimeSPEC predicts $29.5M (28.2% relative error). A naive comparison would conclude that the first three methods vastly outperform TimeSPEC. Shapley-DCLR reveals why this conclusion is misleading.

Temporal Hint's rationale contains five claims, four of which are temporally valid: Durant's free-agent status, the Nets' pursuit, the CBA maximum salary for a 10+-year veteran (approximately $39.9M), and Durant's 11 years of service. The single leaked claim (C7, Shapley rank 1, $\phi = 0.222$) states that "reports in late June 2019 indicated that Kevin Durant agreed to a 4-year, $164 million contract with the Brooklyn Nets"—the target outcome itself, with a verified source date of June 30, seven days after t_{ref}. This claim carries the highest Shapley value among all five claims, indicating that the model's near-perfect prediction is anchored by memorized knowledge of the actual signing. DCLR = 22.2%: over one-fifth of the decision weight traces to this single leaked fact.

Self-Correction produces identical results (DCLR = 22.3%) because the LLM-only supervisor fails to catch the leaked claim. The model backdated the source to "late June 2019" without specifying a precise date, and without external search the supervisor cannot distinguish pre-cutoff rumors from the post-cutoff announcement. This illustrates a limitation of LLM-only verification: when the temporal margin is narrow—seven days in this case—the supervisor lacks the independent evidence needed to detect subtle backdating.

Temporal RAG doubles the contamination. Its retrieval pipeline surfaces two explicitly post-cutoff claims: "on June 30, 2019, Kevin Durant announced that he planned to sign with the Brooklyn Nets" ($\phi = 0.111$) and "Durant signed with Brooklyn on July 7 on a four-year, $164.3 million contract, in a sign-and-trade deal" (Shapley rank 1, $\phi = 0.348$). The latter claim carries the highest decision impact by a wide margin—it effectively provides the answer as retrieved evidence, explaining the near-zero prediction error. DCLR rises to 45.8%. The two safe claims (CBA salary rules and Durant's championship credentials) carry lower Shapley weight, confirming that the prediction is driven primarily by leaked contract details rather than market-rate inference.

TimeSPEC's search-escalated verification correctly identifies and blocks all post-cutoff contract details. The validated evidence grounds the prediction in CBA maximum-salary rules and Durant's service-year eligibility, yielding $29.5M—the approximate CBA maximum for a 10+-year veteran without supermax eligibility. The 28.2% gap between this estimate and the actual $41.06M AAV reflects the difference between what CBA rules predict and the above-maximum deal Durant negotiated. The key point is that TimeSPEC's "error" is honest: the prediction is fully grounded in pre-cutoff evidence (DCLR = 0.0%), while the three other methods' accuracy is partially explained by access to the actual contract value. Shapley-DCLR makes this distinction interpretable: it identifies the specific leaked claim (the contract announcement) and quantifies its contribution to the prediction (22–35% of decision weight depending on the method), rather than merely flagging the instance as contaminated.

E.3 Legal Case Prediction: Hargress v. SSA (2018)

Joyce Hargress appealed the denial of her Social Security disability benefits to the U.S. Court of Appeals for the Eleventh Circuit. The ALJ had found that Hargress, who alleged disability due to type II diabetes, tiredness, and anxiety, could perform sedentary unskilled work. The Appeals Council denied review. The reference date is $t_{\text{ref}} =$ December 29, 2017; the court affirmed the denial on February 27, 2018, in a unanimous opinion (respondent prevails, $y = 0$).

Temporal Hint and Temporal RAG both predict $\hat{p} = 0.00$ (perfect accuracy), while Self-Correction and TimeSPEC both predict $\hat{p} = 0.15$ (slightly lower but still high accuracy). Shapley-DCLR again inverts the apparent ranking.

Temporal Hint achieves DCLR = 100%: every one of its five extracted claims describes the court's decision. The first claim (C6) states that "the case was decided by the Eleventh Circuit on January 11, 2018," while the remaining four (all C8) describe the court affirming the denial, finding substantial evidence to support the ALJ, and reject-

ing Hargress's arguments. Each effective source date falls in January 2018—two weeks after t_{ref}. The five Shapley values are uniform ($\phi = 0.200$ each), and since every claim is leaked, DCLR = 100%. The model's perfect prediction ($\hat{p} = 0.00$) is not forecasting—it is recall of the known outcome. Shapley-DCLR correctly identifies that the entire decision-driving reasoning chain is contaminated.

Self-Correction detects and removes all five leaked claims. The supervisor correctly classifies them as post-cutoff (C6 and C8 categories are deterministically leaked under the taxonomy), triggers regeneration, and the model produces five replacement claims—all temporally valid. Two C0 claims describe the standard of review for Social Security disability appeals and the Eleventh Circuit's historically deferential posture toward ALJ findings. Two C1 claims summarize the ALJ's residual functional capacity determination and the Appeals Council's denial of review. All source dates fall between 2015 and 2017, well before t_{ref}. The prediction shifts from $\hat{p} = 0.00$ to $\hat{p} = 0.15$: the model assigns a low but nonzero probability to the petitioner, reflecting legitimate analytical uncertainty about whether the ALJ's credibility determination survives appellate review. DCLR = 0.0%.

Temporal RAG reintroduces contamination. Its retrieval pipeline surfaces court records containing the outcome: three of four claims are leaked (DCLR = 78.8%), including the highest-Shapley claim ($\phi = 0.288$), which states that "the Eleventh Circuit affirmed the denial" with a January 2018 source date. The single safe claim describes the ALJ's 2014 residual functional capacity finding. The model predicts $\hat{p} = 0.00$ with certainty derived from the retrieved ruling.

TimeSPEC blocks all post-cutoff court records through search-escalated verification. The validated evidence consists of the procedural record available before the appellate decision, and the prediction ($\hat{p} = 0.15$) matches Self-Correction's output at DCLR = 0.0%.

This case illustrates an important edge case for the legal task. On average, legal prediction exhibits the lowest DCLR across all methods (Section 4.2), because most instances are predicted from precedent and procedural history established years before the decision. This specific instance, however, shows that individual legal cases can exhibit extreme contamination when the model directly recalls the ruling. The distinction between $\hat{p} = 0.00$ (contaminated certainty from recalled outcome) and $\hat{p} = 0.15$ (legitimate uncertainty from pre-cutoff procedural analysis) is precisely what Shapley-DCLR is designed to detect: not merely whether leakage is present, but whether it drives the prediction. In this case, it drives the entire prediction—and the evaluation framework correctly reports DCLR = 100%.

F Prompt Specifications

All prompts are reproduced from the implementation source code for reproducibility. Template variables are shown as `{variable}`. Section F.1 covers generation prompts shared across the four methods; Section F.2 covers supervision and regeneration prompts used by Self-Correction and TimeSPEC; Section F.3 covers evaluation pipeline prompts. Table 8 summarizes which prompts each method uses.

Table 8: Prompt usage across methods. ✓ = used; – = not used.

Prompt	T. Hint	Self-Corr.	T. RAG	TimeSPEC
Generation (System + User)	✓	✓	✓	✓
RAG Context Block	–	–	✓	✓
Query Generation	–	–	✓	✓
Taxonomy Verification	–	✓	–	✓
Search Re-Verification	–	–	–	✓
Regeneration	–	✓	–	✓
Aggregation	–	✓	–	✓

F.1 Generation Prompts

All four methods share the same generation backbone. The system prompt embeds a strict temporal constraint and a three-step evidence–reasoning–prediction procedure. The user prompt supplies the question and optional RAG context. Task-specific output schemas are appended to the system prompt.

Shared System Prompt. Used by all four methods (B1–B4). Temporal RAG and TimeSPEC receive additional RAG context in the user message; the system prompt is identical.

Generation System Prompt

You are an expert forecaster. Your task is to answer a prediction question using only information that was publicly available before the specified knowledge cutoff date.\n\n STRICT TEMPORAL CONSTRAINT:\n You may ONLY use information that was publicly available BEFORE `{cutoff_date}`. Do NOT use knowledge of events, announcements, or developments from on or after `{cutoff_date}`. If uncertain whether a piece of information predates the cutoff, omit it.\n\n EVIDENCE INTEGRITY:\n Evidence must be factually genuine. Do not fabricate numbers, dates, or events. Each evidence item must reflect a real, publicly known fact. If you are uncertain about a specific value, state the fact qualitatively rather than inventing a precise figure.\n\n You MUST follow this three-step procedure:\n\n STEP 1 — EVIDENCE:\n List every verifiable fact you will use. Requirements:\n - Each fact must be concrete and specific — not an opinion or prediction.\n - Each fact must have been publicly known before `{cutoff_date}`.\n - For EACH fact, include a source_date field: the YYYY-MM-DD date when this information was FIRST PUBLICLY ANNOUNCED.\n\n STEP 2 — REASONING:\n Write a reasoning paragraph that cites evidence by number (e.g. [1], [3]).\n - You MUST cite every evidence item at least once.\n - Do NOT introduce any new facts not listed in your evidence.\n\n STEP 3 — PREDICTION:\n Give your final prediction based solely on the reasoning above.\n CRITICAL: You MUST always output the JSON object below, even under uncertainty.\n `{output_spec}`

User Message Structure. The user message is constructed programmatically. For Temporal Hint and Self-Correction, it contains only the question and background. For Temporal RAG and TimeSPEC, a `RETRIEVED CONTEXT (all pre-cutoff)` block is prepended with search results assembled by the RAG pipeline.

Generation User Message

[If RAG context available:]\n RETRIEVED CONTEXT (all pre-cutoff):\n`{context_block}`\n\n QUESTION: `{question}`\n\n BACKGROUND: `{background}`\n\n *[If stock task:]* TICKERS TO RANK: `{tickers}`

Task-Specific Output Schemas. Appended to the system prompt via `{output_spec}`. Each schema enforces structured JSON output with an evidence array, reasoning paragraph, and task-specific prediction field.

Output Schemas (abridged)

Legal:\n {"evidence": [{"id": 1, "fact": "...", "source_date": "YYYY-MM-DD"}],\n "reasoning": "...", "probability_petitioner": ⟨float in [0, 1]⟩}\n\n **Salary:**\n {..., "predicted_salary": ⟨positive integer; AAV in USD⟩}\n\n **Stock:**\n {..., "ranking": ["TICKER_BEST", ..., "TICKER_WORST"]}\n\n **Binary (large-scale ablation):**\n {..., "probability_yes": ⟨float 0.0–1.0⟩, "prediction": "yes/no"}

RAG Query Generation. Used by Temporal RAG and TimeSPEC to generate search queries for the retrieval pipeline. The prompt is user-only (no system message).

RAG Query Generation Prompt

You are generating web search queries to gather relevant evidence for a forecasting question. All evidence must predate the cutoff date.\n\n CUTOFF DATE: `{cutoff_date}`\n\n QUESTION: `{question}`\n\n BACKGROUND: `{background}`\n\n Generate between `{min_queries}` and `{max_queries}` search queries.\n - Each query should target a DISTINCT aspect of the information you need.\n - Include the year in each query to anchor temporal relevance.\n - Do not include queries about events after `{cutoff_date}`.\n\n Output as JSON array: [{"query": "..."}]

F.2 Supervision and Regeneration Prompts

Self-Correction and TimeSPEC share the supervision–regeneration–aggregation loop. The taxonomy verification prompt is also shared with the evaluation pipeline (Section F.3).

Taxonomy Verification. Used by the generation-time supervisor (Self-Correction, TimeSPEC) and the post-hoc evaluation pipeline. The system prompt defines the auditor role; the user prompt supplies claims and the C0–C8 decision tree.

Taxonomy Verification System Prompt

You are a temporal provenance auditor.\n\n Your task is public-availability date verification, not factual truth checking and not outcome prediction. For each claim, decide the earliest reliable date on which the claim could be publicly verified. Use this date to determine leakage relative to the cutoff. Return only valid JSON matching the requested schema.

Taxonomy Verification User Prompt

Cutoff date: `{cutoff_date}`\n Task: `{task}`\n Target outcome description: `{target_description}`\n\n Apply the C0-C8 decision tree independently to each item. Stop at the first matching rule.\n\n For each item:\n 1. Treat declared_source_date as a candidate, not ground truth.\n 2. Decide whether declared_source_date is plausible for when this claim could first be publicly verified.\n 3. If implausible, provide the earliest reliable corrected date.\n 4. If uncertain, use the conservative latest plausible date and lower confidence.\n 5. Do not label a stable/background fact leaked merely because it appears in later litigation.\n 6. For claims about a specific case order, holding, filing, grant, denial, oral argument, or opinion, use the public release date.\n 7. Leakage: effective_source_date > cutoff_date, except C7/C8 which are leaked by definition.\n\n Items: `{items_json}`\n\n Output JSON: {"results": [{"id": 1, "category": "C0", "declared_date_plausible": true, "effective_source_date": "YYYY-MM-DD", "is_leaked": false, "confidence": "high", "reason": "..."}]}

Search Re-Verification (TimeSPEC only). After the initial taxonomy pass, claims with uncertain or borderline dates are escalated to Perplexity search. The search snippets and original claim are then passed to this prompt for re-verification.

Search Re-Verification Prompt

System: You are a temporal provenance auditor. You have been given a claim and search results that were retrieved to verify the public availability date of that claim.\n\n Based on the search results, determine the earliest reliable date on which the information in the claim would first have been publicly knowable.\n\n Return only valid JSON: {"effective_source_date": "YYYY-MM-DD or null", "is_leaked": true/false, "confidence": "high|medium|low", "reason": "brief explanation"}

User: Claim: `{claim_text}`\n Declared source date: `{declared_date}`\n Cutoff date: `{cutoff_date}`\n\n Search results:\n`{search_snippets}`\n\n Based on these search results, what is the earliest reliable public availability date for this claim? Return JSON only.

Regeneration. When the supervisor detects leaked claims, the regeneration prompt prepends a leakage warning to the shared generation system prompt. The user message includes the question, optional RAG context, and the previous evidence list for reference.

Regeneration System Prefix (prepended to Generation System Prompt)

The following evidence items from your previous response were flagged as potentially containing information that was not publicly available before the cutoff date: `{leaked_ids}`\n\n You must generate a new response that does NOT rely on these items. Build a fresh evidence list and reasoning that avoids the leaked information. Your evidence list may include other facts, but not facts that depend on or refer to the flagged information.

Aggregation. When any leakage was detected across generation cycles, the aggregator synthesizes a final prediction from only the deduplicated validated (non-leaked) evidence. This enforces the closed-world constraint.

Aggregator Prompt

System: You are an expert forecaster. You have been given a list of verified evidence items that have been confirmed to predate the knowledge cutoff. Using ONLY this evidence, produce a final structured prediction. Do not introduce any new facts not in the provided evidence list.

User: Evidence items (all verified as pre-cutoff):\n`{evidence_json}`\n\n QUESTION: `{question}`\n\n BACKGROUND: `{background}`\n\n `{output_spec}`

F.3 Evaluation Pipeline Prompts

Claim Extraction (Phase 1). Claims are extracted as structured JSON during generation: each method produces an `evidence` array of `{id, fact, source_date}` objects as part of its output schema. The evaluation pipeline loads these directly without a separate LLM extraction call. Taxonomy classification uses the verification prompt above (Section F.2).

Shapley Coalition Prediction (Phase 2). For each coalition subset, the model predicts from only the included claims. The system prompt is stable per instance (cacheable); the user prompt varies per coalition. Task-specific prompts are shown below.

Shapley Coalition — Legal

System: You are a probability estimator. Output ONLY valid JSON (no markdown, no prose). The JSON object must contain only the key probability_petitioner — nothing else.

User: Predict P(petitioner wins).\n Case: `{case_name}` | Petitioner: `{petitioner}` | Respondent: `{respondent}`\n Evidence: `{evidence_json}`\n Reply with exactly: {"probability_petitioner": ⟨float 0.0-1.0⟩}

Shapley Coalition — Salary

System: You are an estimator. Output ONLY valid JSON, no other text.

User: Q: `{question}`\n Facts:\n`{numbered_facts}`\n Output schema: {"salary": positive_number}

Shapley Coalition — Stock

System: You are a stock analyst. Output ONLY valid JSON (no markdown, no prose). The JSON object must contain only the key ranking — nothing else.

User: Rank tickers by expected return.\n Allowed: `{ticker_universe_json}`\n Evidence: `{evidence_json}`\n Reply with exactly: {"ranking": ["TICKER1", "TICKER2", ...]}

Shapley Coalition — Binary

System: You are a probability estimator. Output ONLY valid JSON (no markdown, no prose). The JSON object must contain the single key probability_yes — nothing else.

User: Estimate P(YES) for: `{question}`\n Evidence: `{evidence_json}`\n Reply with exactly: {"probability_yes": ⟨float 0.0-1.0⟩}

Leakage Verification (Phase 3). For claims in categories C2–C6 that require date verification, the taxonomy verification prompt (Section F.2) serves as the primary verification mechanism. Claims with implausible or uncertain dates are escalated to search-based verification via Perplexity API, followed by the search re-verification prompt.