Rillor

METHODS NOTE

Look-ahead bias in LLM backtests and how to test for it

A language model can already know the outcomes in a historical backtest. Published tests detect this look-ahead bias, and point-in-time models and dated inputs reduce it.



Rillor

Contents

01 Summary

02 Background

03 Method

04 Findings

05 Implications

06 Limits of this analysis

07 Sources

Summary

Look-ahead bias in an LLM backtest is the use of information that was not available at the forecast date, and a language model can carry that information in its training data as well as in its inputs [9, 18].

- A language model trained on text from a backtest's period can already know the outcomes it is asked to forecast. Before its training cutoff, one widely used chat model recalled exact index levels and economic figures, and instructions to ignore later information did not stop the recall [2].
- Look-ahead bias reaches an LLM backtest through the model's parameters and through the documents it retrieves [9]. Date-restricted search is unreliable, and a model's stated knowledge cutoff is an approximation [4, 5].
- Hiding company names and dates cuts both ways. In one sentiment study, anonymized headlines did better within the training window because company knowledge distracted the model [1]. Other work finds that models reconstruct masked names and dates [2] and that masking removes useful information [17].
- Lookahead Propensity asks a model about a firm and a date with no text at all and measures how sure it is of the outcome. In its authors' tests it was materially positive on dates before the training cutoff and close to zero after it [3].
- Lower scores after a model's training cutoff do not prove leakage before it, because models know more about times near their cutoff. Four of five flagship models failed this check on questions that all resolved after their cutoffs [4].
- Point-in-time language models are trained only on text available up to a date. Two recent families have 1.3 billion and up to 4 billion parameters [7, 8], and the authors of the second report a gap to unrestricted models on several tasks [8].
- A backtest's inputs also need the values known at each forecast date. The St. Louis Fed's FRED data API returns today's view of revised data by default, and a company filing carries its own acceptance time [11, 26].
- A backtest that does not control for the number of trials behind it gives over-optimistic performance expectations [20]. Applied to an LLM backtest, every prompt, model and setting tried counts as one trial.

Background

Look-ahead bias before language models

The term comes from backtesting with data. A learning note from the FISD Alternative Data Council puts the requirement in one sentence: "If the model is to accurately predict the future from the past, the history described by the data needs to reflect, at each Point-In-Time, what was known and when it was known." [18]. It names look-ahead bias as the result of using data from after the date being analyzed, and survivorship bias as the result of a company universe that represents only the present [18]. A 2026 preprint that reviews language models in finance defines look-ahead bias as when "The model accidentally uses future information that would not have been known at the time a decision was supposed to be made." [9].

Macroeconomic series are revised after release, and the Federal Reserve Bank of St. Louis runs ALFRED, which "allows you to retrieve each economic data release (vintage) that was available on a specific date in history" [10]. The Federal Reserve Bank of Philadelphia's Real-Time Data Set for Macroeconomists "consists of vintages, or snapshots, of time series of major macroeconomic variables", and its page names Croushore and Stark's 2001 paper in the Journal of Econometrics as the preferred citation [22].

What language models add

Paleka and colleagues define backtesting for language model forecasters: "the forecasting system is given knowledge as of some past time T and asked to forecast events between time T and the present" [5]. A language model brings its own knowledge to that setup. Glasserman and Lin write that a backtest is biased "if the training and backtesting periods overlap" [1].

Early studies avoided the overlap by design. Lopez-Lira and Tang's study of headline-based return prediction, first posted in April 2023, runs from October 2021 to May 2024 on 4,123 companies, with a dated model version whose training data stops in September 2021 [6]. Lopez-Lira, Tang and Zhu later state the general problem: "Counterfactual forecasting ability is non-identified when the model has seen the realized values: any observed output is consistent with both genuine skill and memorization." [2].

How often studies address it

Kong and colleagues reviewed 164 papers from 2023 to 2025 that evaluate language models in finance, and no single bias they track is discussed in more than 28 percent of them [9]. Only 26.8% (44 papers) acknowledge look-ahead bias, and survivorship bias appears in 1.2% (2 papers) [9]. In the same preprint's user study, 74% of practitioners said tools for bias evaluation were scarce (48%) or non-existent (26%), and 50% named this as the primary bottleneck [9].

Method

What was examined. Published designs and tests for finding and reducing look-ahead bias when a language model forecasts outcomes or scores text about past periods. Each is described from the paper or agency page that sets it out, with what it needs and what its authors say it cannot do.

Sources. Twenty-eight sources, each opened on 8 November 2026 and read for the passages cited. Nine are preprints from 2025 and 2026 with no journal reference on arXiv [2, 3, 4, 5, 7, 8, 9, 16, 17]. Eight are peer-reviewed or accepted papers [1, 12, 13, 15, 20, 21, 23, 25], three are other arXiv papers [6, 14, 24] and one is the authors' revised working paper [19]. Four are pages and documents from the Federal Reserve Banks of St. Louis and Philadelphia and the U.S. Securities and Exchange Commission [10, 11, 22, 26]. The rest are an industry learning note [18], a dataset card [27] and a benchmark's website [28]. The venues of [1], [12], [21] and [23] come from the reference list of [4].

How. Several sources study return prediction. This note treats those returns as research subjects: it reports what each paper found about leakage and leaves out returns, ratios and hit rates. Where a paper names the model it tested, this note describes the model instead. Every figure is the source's own.

Where sources disagree. The abstract page of Kelly and colleagues' preprint (version 3) describes monthly checkpoints, while the paper's text describes yearly checkpoints from 2013 to 2024 and year-end releases [8]. This note follows the paper's text. Kelly and colleagues and the FineWeb dataset card describe FineWeb's dates differently, and this note uses the card's definition, the crawl date [8, 27]. Halawi and colleagues give two different knowledge cutoff dates in one version of their paper, so this note cites only their retrieval design [12].

Findings

1. Two routes for leakage: model parameters and retrieved documents

Kong and colleagues write that look-ahead violations "occur through two distinct channels: (1) parametric knowledge embedded in model weights, and (2) incorporation of external context via retrieval-augmented generation (RAG)" [9]. Declaring a knowledge cutoff fails because "Cutoffs are opaque and not verifiable". Filtering retrieved documents by publication time fails because "Retrieved documents are edited and re-ranked using future information" [9].

Logical leakage. Paleka and colleagues add a path that needs neither weights nor search: "Knowledge of the backtesting date can leak outcomes." [5]. A 2021 question on whether Queen Elizabeth would live to 100, backtested in 2025, can only resolve "no" [5]. They report that "at least 3.8% of their dataset consists of questions for events that resolved 'early'" in one published question set and that at least 10% of another set's questions were trivial to forecast [5]. Their fix: "Forecasting questions collected for backtesting should be restricted to those where every possible resolution of the question can be validated at the time of evaluation." [5].

Retrieval leakage. "Many search engines do not robustly implement date restrictions." [5]. Pages are updated "without changing their reported publication date", and "the retrieval engine may rely on algorithms or knowledge developed after the evaluation time T" [5]. In their example, a search for "January 6th" restricted to 2020 "returns documents with an abnormally strong association to U.S. politics (by 2020 standards)" [5]. They call a filter that discards articles with later information "imperfect" [5]. Their more robust options are a retrieval system "which only returns results from Wikipedia and news sources with clear and reliable dates", or an index kept for each time period [5]. Against ranking leakage they suggest "a simple TF-IDF system" or "older embedding models", at the cost of "much worse retrieval performance" [5].

Dated retrieval in practice. Halawi and colleagues retrieve "articles between the question begin date and a specified retrieval date", a date that "can be viewed as the 'simulated date' of the forecast" [12]. Each question gets five retrieval dates at geometric intervals between its open and close dates, and dates after an early resolution are dropped [12]. Choosing retrieval dates from the resolve date "would leak information, since the retrieval date would now depend on the resolve date, which ... correlates with the resolution" [12].

Scaffolding and human forecasts. System prompts "and other scaffolding of the model can leak information" [5]. Human forecasts "may be contained in training data or retrieval system responses", so a claim that a model matches human forecasters "may then be circular" [5].

2. Evidence of memorization before the cutoff

Lopez-Lira, Tang and Zhu tested a widely used chat model with an October 2023 training cutoff in a preprint last revised in December 2025 [2]. Before that cutoff, the model "can recall specific S&P 500 index levels with perfect precision on certain dates, unemployment rates accurate to a tenth of a percentage point, and precise quarterly GDP growth figures" [2].

Instructions did not stop it. Told to ignore information after 2010 when forecasting the direction of quarterly GDP growth, the model reached "97.6% threshold accuracy before the artificial cutoff and 98.0% after", while its "actual post-knowledge cutoff accuracy is only 40%" [2]. Masking did not stop it either. With company names, numbers, locations and dates removed from earnings call transcripts, it still identified the company in all calls of three large technology firms and in above 85% of the calls for each of seven, and one firm's quarter and year with 93.2% accuracy [2]. After the cutoff, the authors "observe no recall" [2]. They add that "Any evidence of memorization represents only a lower bound on encoded knowledge." [2].

Gao, Jiang and Yan give one case in a preprint revised in June 2026 [3]. For a firm whose stock jumped on 29 July 2020, the day after a loan announcement on 28 July 2020, an open-weight model with a December 2023 training cutoff gave "Pup ≈ 0.9999" to a query with only the firm's name, ticker and that date, "no headline at all" [3].

3. Anonymization and the distraction effect

Three studies of anonymization find three different effects [1, 2, 17].

It can improve in-sample results. Glasserman and Lin separate "a look-ahead bias, in which the LLM may have specific knowledge of the stock returns that followed a news article, and a distraction effect, in which general knowledge of the companies named interferes with the measurement of a text's sentiment" [1]. They scored 129,431 headlines on 6,723 companies and 181,908 headlines on 678 S&P 500 companies, in sample from January 2015 to September 2021, when the model's training data ends, and out of sample from October 2021 to December 2022 [1]. "In-sample (within the LLM training window), we find, surprisingly, that the anonymized headlines outperform, indicating that the distraction effect has a greater impact than look-ahead bias." [1]. The tendency is "particularly strong for larger companies", and out of sample "look-ahead bias is not a concern but distraction remains possible" [1].

It can leak. Lopez-Lira, Tang and Zhu find that "masking fails as LLMs reconstruct entities and dates from minimal context" [2].

It can remove information. Wu, Yang, Ying and Zhou, in a preprint revised in September 2026, predict S&P credit downgrades from raw and anonymized texts [17]. "Raw texts significantly outperform anonymized texts." [17]. Look-ahead bias does not explain the difference, they write, and anonymization instead "degrades predictive performance by masking informative numerical and agent entities" [17]. The loss spans "various LLMs, tasks, and text types" [17]. They propose an anonymization gap, the attenuation-adjusted absolute difference between raw and anonymized scores, as "a metric requiring no outcome data" [17].

Read together, a gap between raw and anonymized results mixes look-ahead bias, distraction and lost information [1, 17], and a masked text can still be identified [2]. Anonymization changes the input, so its result needs a test of its own.

4. Detecting contamination: Lookahead Propensity

Gao, Jiang and Yan's test asks what the model knows about an outcome without showing it the text. "Using a date-only recall query for a firm-date pair, we estimate the probability that the LLM has internalized information about the realized outcome, a statistic we term Lookahead Propensity (LAP)." [3]. The query "contains the firm's name, ticker, and target date and nothing else: no headline, no transcript" [3]. LAP is the probability the model places on knowing the outcome, P(up) plus P(down), against an option that it does not know [3].

What it found. "LAP is materially positive throughout the in-sample period and collapses essentially to zero right after the training-data cutoff." [3]. On stock news in sample, "roughly 42% of firm-days place essentially all mass on P(unknown), and roughly 23% are saturated at LAP P(known) ≥ 0.95" [3]. Across all 7,806 firm-day queries in 2024, after the cutoff, the maximum LAP was below 0.0001 [3]. On earnings calls in sample, only 2.1% of firm-quarters lie above 0.5, and after the cutoff the maximum across 6,619 queries is below 0.00001 [3].

The regression test. LAP is joined to the forecast in an accuracy regression, and "a positive interaction between LAP and the LLM forecast in an accuracy regression indicates lookahead-bias contamination" [3]. The authors apply it to headlines predicting stock returns and to earnings call transcripts predicting capital expenditures [3]. In the capital expenditure task "the LAP interaction is positive and significant (t = 2.01)" and "collapses to insignificance on a post-cutoff window" [3].

What it needs. A panel of texts with matched realized outcomes, the model's training cutoff date, "typically disclosed in the model card", and "access to token-level probabilities at the answer position, which open-weight models provide directly and some commercial APIs expose as log-probabilities" [3]. Without token probabilities, they "can be approximated, more coarsely, by answer frequencies across repeated samples at positive temperature" [3].

Why the authors built it. They call earlier fixes "useful but partial": masking leaves "subtler memorization channels intact", and post-cutoff windows for popular models are short, "which limits statistical power" [3].

Tests from static benchmarks. Other detectors ask a different question. Golchin and Surdeanu's guided instruction gives the model a dataset's name, partition and the start of an instance and flags it "if the LLM's output either exactly or nearly matches the latter segment of the reference" [13]. It agreed with human experts with "an accuracy between 92% and 100%" on seven datasets [13]. Shi and colleagues' Min-K% Prob assumes that "an unseen example is likely to contain a few outlier words with low probabilities under the LLM", and needs no knowledge of the pretraining corpus [14]. Both ask whether a text was seen. LAP asks whether an outcome is known [3]. Zhang and Stadie describe backtest leakage as "an indirect information flow through a common cause of the training text and the outcome, not verbatim overlap", and note that "Detection methods establish that contamination exists, not what it is worth" [4].

5. Why before-and-after-cutoff comparisons mislead

The common check compares a model's backtest score before its training cutoff with its score after. Zhang and Stadie test it in an August 2026 preprint [4]. "We show this check is uninformative. Four flagship models fail it on questions they cannot have memorized: every scored question resolved after their cutoffs." [4]. On questions that resolved at least 30 days after each model's documented cutoff, four of five flagship models showed "spurious 'leakage' gaps as large as +0.061" in improvement over the crowd [4]. The data are 1,646 resolved binary questions from four prediction-market and forecasting platforms, resolving from the third quarter of 2024 through the third quarter of 2026, scored as a "crowd-anchored Brier reduction" [4].

Recency mimics leakage. A backtest score "mixes three components", they write: skill, recency, which "is legitimate knowledge that varies with distance from the cutoff", and leakage, which "is memorized outcome information" [4]. "Models legitimately know more about times near their cutoff, so recency mimics leakage, and we prove no passive backtest can separate the two from genuine skill." [4]. Older work shows the decay: language models "perform worse in the realistic setup of predicting future utterances from beyond their training period, and that model performance becomes increasingly worse with time" [15]. The reverse reading fails too: "A flat pre/post profile is not evidence of a clean backtest." [4].

Leakage is uneven. It "concentrates on outcomes that surprised the crowd and were well covered in training, and partial memorization is disproportionately rewarded" [4]. In their analysis, half the leaked information already yields three quarters of the full score inflation, so "an average contamination rate is the wrong audit object" [4].

One defensible reference. A known cutoff identifies leakage at the boundary, and a matched clean control identifies it across the whole period and "yields a leakage-adjusted score" [4]. At the boundary, the test is a regression discontinuity: "honest competence drifts continuously through time, while memorization switches off at the cutoff. Any jump in the loss there is therefore leakage." [4]. For the adjusted score, "families ship vintages with different cutoffs, so clean controls are standard artifacts, selected rather than retrained, and they need to be clean only on the evaluation window" [4]. The control has to share the target's recency profile, and "a control too weak to express recency overstates" the inflation [4]. Backtests, they conclude, "need not be discarded" and "need one defensible reference" [4]. Their guarantees "are conditional on assumptions made explicit", and "documented cutoffs are month-resolution" [4].

Small samples after the cutoff. Lopez-Lira, Tang and Zhu reach a related result: "When post-cutoff sample size is small relative to pre-cutoff sample size, genuine forecasting skill and undetected memorization cannot be reliably distinguished by hypothesis tests." [2]. Where post-cutoff data are large enough, "they should be the primary analysis, not a robustness check" [2].

The cutoff itself. Paleka and colleagues write that "cutoffs are more of a guideline than a guarantee", and that the term "may refer only to the pretraining data" [5]. A widely used chat model with a stated October 2023 cutoff, told in its system prompt that its cutoff was November 2023, described a meeting first announced on

8 November 2023, though the authors say they cannot confirm this as leakage, since the visit had been discussed in October 2023 [5]. A model's release date, they write, is a strict upper bound on its knowledge cutoff, except where the system prompt is updated [5].

Static benchmarks differ. LiveCodeBench tags each coding problem "with a release date" and found "a stark drop in the performance" of three models on problems released after their cutoffs [23]. Retro-holdouts apply a clean control to a fixed benchmark: a dataset built to be statistically indistinguishable from TruthfulQA showed that some of twenty models "have inflated scores by as much as 16 percentage points" [24]. Zhang and Stadie argue that such estimates "do not transfer, because a backtest defines 'clean' by a date rather than by membership" [4].

Forecasting live. ForecastBench "is comprised solely of questions about future events that have no known answer at the time of submission", and its question bank is updated by "an automated process that runs nightly at 0:00 UTC" [21]. Its website calls it "a dynamic, contamination-free benchmark" with human comparison groups [28]. Its questions are the panel Zhang and Stadie audit [4].

6. Point-in-time models trained to a date

Kelly and colleagues describe point-in-time models as "trained exclusively on text available up to each calendar date" and note that earlier efforts "typically produce models that lag substantially behind their unconstrained counterparts" [8].

A family with annual cutoffs. Yan and colleagues, in a 2026 preprint, trained "a family of twelve 1.3B-parameter language models trained from scratch on approximately 100 billion tokens each with strict annual data cutoffs spanning 2013 to 2024" [7]. They also built an instruction dataset grounded in each year's documents, because responses written later carry knowledge past the cutoff "that is difficult to detect and filter" [7]. Their perplexity probing "confirms that each model's knowledge is bounded by its cutoff year" [7].

Scaling up. Kelly, Malamud, Schwab and Xu trained models of "up to 4 billion parameters on 1 trillion chronologically filtered tokens from FineWeb", with year-end checkpoints and a latest release dated December 2024 [8]. Their models approach open-weight models of comparable size trained on unrestricted data, "although a performance gap remains on several tasks" [8]. On PIQA and WinoGrande the gap "is nearly negligible" [8].

Earlier work. He, Lv, Manela and Wu built "a pair of models for each year starting in 1999", with cutoffs from 1999 to 2024 and data "carefully filtered by publication date" [16]. Drinkall and colleagues trained small yearly models from 2011 to 2022, "specifically designed to be nonprognosticative" [25].

How much it matters depends on the task. He and colleagues write that "Lookahead bias is model and application-specific", because a prediction model on top of a weaker point-in-time model "can compensate" [16]. In their news-based return application, "lookahead bias is modest" [16]. Gao and colleagues found LAP materially positive in sample on their tasks [3], and Zhang and Stadie found leakage concentrated on surprising outcomes [4].

What a date means. Yan and colleagues filter by crawl date and note that "Crawl timestamps do not reflect the true creation date of a webpage, as a page crawled in 2015 may have been authored in the 1990s" [7]. Filtering by crawl date still ensures "that a model is never exposed to data crawled after its designated cutoff" [7]. The FineWeb dataset card defines its date field as "crawl date (from CommonCrawl)" [27].

7. Inputs need vintages too

The FRED API documentation explains that "observation data values can be revised" and that "The real-time period marks when facts were true or when information was known until it changed." [11]. "On almost all URLs, the default real-time period is today." [11]. ALFRED users "can change the real-time period to retrieve information that was known as of a past period of history" [11]. A call left at the default returns revised history [11]. The Philadelphia Fed's data set offers the "Complete vintage history" or only the first three releases [22].

Company data revise too. The SEC's Financial Statement and Notes data sets hold data "'as filed' by the registrant", including "amendments of prior submissions" [26]. Each submission carries `accepted`, "The acceptance date and time of the registrant's filing with the Commission", and "Filings accepted after 5:30pm EST are considered filed on the following business day." [26]. The field `prevrpt` marks a submission whose information "was subsequently amended prior to the end cutoff date of the data set" [26]. Paleka and colleagues, citing earlier work, note that "Enron drastically changed its 1998-2001 earnings in 2002, and many data repositories do not report the original values anymore" [5].

The company universe needs a date as well. FISD names survivorship bias for a universe that "only represents the present" [18], and Kong and colleagues' checklist pairs "Point-in-Time Data Access" with a dynamic universe [9].

8. Overfitting compounds leakage

Searching over many variants is a second way a backtest overstates skill. Bailey, Borwein, López de Prado and Zhu write that hold-out and other standard techniques against overfitting "tend to be unreliable and inaccurate in the context of investment backtests", and propose "the probability of backtest overfitting (PBO)", computed by "combinatorially symmetric cross-validation (CSCV)" [19]. "Backtests published in academic or practitioners' publications almost never declare the number of trials involved in a discovery." [19]. They add that "as long as the researcher tries more than one strategy configuration, overfitting is always present" [19].

Bailey and López de Prado's deflated Sharpe ratio corrects for "Selection bias under multiple testing and non-Normally distributed returns" [20]. They write that the number of trials attempted is missing from virtually all backtests published in academic journals and investment offerings, and that a backtest without control for the extent of the search "is worthless" [20].

Applied to an LLM backtest, a trial in the sense of [19, 20] is any variant tried: a prompt wording, a model, a retrieval setting or an input set. Leakage and search each inflate the score, and each needs its own correction: a reference for leakage [4] and a count of trials for the search [19, 20].

Implications

A checklist for an LLM backtest. Each step traces to the sources above.

1. Write down the as-of time of every forecast and fetch every input as of that time: data vintages and real-time periods [10, 11, 22], filing acceptance times [26] and a dated company universe [9, 18].
2. Retrieve from an index with reliable dates, or one kept per period, and treat filters on later articles as imperfect [5, 9]. Never set retrieval dates from the resolution date [12].
3. Record the model version and its documented training cutoff, and treat the cutoff as an approximation [4, 5, 6].
4. Keep only questions whose every resolution could be checked at the as-of time [5].
5. Run a detection test: date-only recall and LAP where token probabilities are available, or answer frequencies across repeated samples where they are not [3].
6. Read neither a flat nor a falling before-and-after split as a result. Use one reference: a known cutoff for leakage at the boundary, or a matched clean control for a leakage-adjusted score [4]. Make post-cutoff data the primary test when there is enough of it [2].
7. If texts are masked, measure both sides: the bias removed and the information lost [1, 2, 17].
8. Where it fits the task, compare with a point-in-time model trained to the as-of date, and expect a capability gap [7, 8, 16, 25].
9. Count every prompt, model and setting tried, and correct for the count [19, 20].
0. Where the question allows it, forecast live on questions with no known answer [21].

For research leads. A historical score from a model trained on the test period mixes skill, recency and leakage [4]. Ask which reference it used [4] and how many trials preceded it [19, 20].

For engineers. The tests need records that only the pipeline can keep: the as-of time of each forecast, the exact inputs and their dates, the model version and, for LAP, the token probabilities [3, 4, 11, 26]. Retrieval needs an index whose dates can be trusted, and its ranking can leak on its own [5, 12].

Rillor's forecasting and backtesting pipelines replay only what was known at each point in time. Each forecast record keeps the exact items each agent saw, with their capture times, and the model family and pinned version behind every agent. An audit agent checks that every claim in the rationale traces to an input that existed at forecast time.

Limits of this analysis

- The evidence on detection and adjustment comes mainly from preprints posted in 2025 and 2026 that show no journal reference [2, 3, 4, 5, 7, 8, 9, 16, 17]. Their methods and results may change in later versions.
- The studies cover U.S. equities, credit ratings, macroeconomic series and prediction-market questions [1, 2, 3, 4, 17]. LAP's authors apply it to stock news and earnings calls only [3], and Zhang and Stadie add a dated coding benchmark [4].
- LAP needs token probabilities, and its authors describe the repeated-sampling substitute as coarser [3].
- Zhang and Stadie's guarantees rest on stated assumptions, and published cutoffs resolve only to the month [4].
- The two recent point-in-time families have 1.3 billion and up to 4 billion parameters [7, 8], and the authors of the second report a gap to unrestricted models on several tasks [8]. Crawl dates bound what they saw, not when a page was written [7, 27].
- Kong and colleagues measured published papers and a user study [9]. No source here measures how often look-ahead bias occurs in industry backtests.
- This note reports no measurement of its own and ranks no model, test or provider.

Sources

1. Assessing Look-Ahead Bias in Stock Return Predictions Generated By GPT Sentiment Analysis. Paul Glasserman and Caden Lin, The Journal of Financial Data Science 6(1), 2024, arXiv 2309.17322, version 1, 29 September 2023, viewed 8 November 2026. https://arxiv.org/abs/2309.17322

2. The Memorization Problem: Can We Trust LLMs' Economic Forecasts? Alejandro Lopez-Lira, Yuehua Tang and Mingyin Zhu, arXiv 2504.14765, version 2, 15 December 2025, viewed 8 November 2026. https://arxiv.org/abs/2504.14765

3. Detecting Lookahead Bias in LLM Forecasts. Zhenyu Gao, Wenxi Jiang and Yutong Yan, arXiv 2512.23847, version 2, 12 June 2026, viewed 8 November 2026. https://arxiv.org/abs/2512.23847

4. Temporal Leakage in LLM Backtesting: Measurement, Validation, and Adjusted Scores. Zeyu Zhang and Bradly C. Stadie, Northwestern University, arXiv 2608.02985, version 1, 4 August 2026, viewed 8 November 2026. https://arxiv.org/abs/2608.02985

5. Pitfalls in Evaluating Language Model Forecasters. Daniel Paleka, Shashwat Goel, Jonas Geiping and Florian Tramèr, arXiv 2506.00723, version 1, 31 May 2025, viewed 8 November 2026. https://arxiv.org/abs/2506.00723

6. Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models. Alejandro Lopez-Lira and Yuehua Tang, arXiv 2304.07619, version 6, 28 October 2025, viewed 8 November 2026. https://arxiv.org/abs/2304.07619

7. DatedGPT: Preventing Lookahead Bias in Large Language Models with Time-Aware Pretraining. Yutong Yan, Raphael Tang, Zhenyu Gao, Wenxi Jiang and Yao Lu, arXiv 2603.11838, version 2, 23 July 2026, viewed 8 November 2026. https://arxiv.org/abs/2603.11838

8. Scaling Point-in-Time Language Models: Enabling Economic Evaluation of Embeddings. Bryan Kelly, Semyon Malamud, Johannes Schwab and Teng Andrea Xu, arXiv 2607.11889, version 3, 12 October 2026 (version 1, 24 April 2026), viewed 8 November 2026. https://arxiv.org/abs/2607.11889

9. Evaluating LLMs in Finance Requires Explicit Bias Consideration. Yaxuan Kong, Hoyoung Lee, Yoontae Hwang and others, arXiv 2602.14233, version 1, 15 February 2026, viewed 8 November 2026. https://arxiv.org/abs/2602.14233

10. ALFRED: Archival Federal Reserve Economic Data. Federal Reserve Bank of St. Louis, viewed 8 November 2026. https://alfred.stlouisfed.org/

11. Real-Time Periods. FRED API documentation, Federal Reserve Bank of St. Louis, viewed 8 November 2026. https://fred.stlouisfed.org/docs/api/fred/realtime_period.html

12. Approaching Human-Level Forecasting with Language Models. Danny Halawi, Fred Zhang, Chen Yueh-Han and Jacob Steinhardt, NeurIPS 2024, arXiv 2402.18563, version 1, 28 February 2024, viewed 8 November 2026. https://arxiv.org/abs/2402.18563

13. Time Travel in LLMs: Tracing Data Contamination in Large Language Models. Shahriar Golchin and Mihai Surdeanu, ICLR 2024, arXiv 2308.08493, version 3, 21 February 2024, viewed 8 November 2026. https://arxiv.org/abs/2308.08493

14. Detecting Pretraining Data from Large Language Models. Weijia Shi, Anirudh Ajith, Mengzhou Xia and others, arXiv 2310.16789, version 3, 9 March 2024, viewed 8 November 2026. https://arxiv.org/abs/2310.16789

15. Mind the Gap: Assessing Temporal Generalization in Neural Language Models. Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya and others, NeurIPS 2021, arXiv 2102.01951, version 2, 26 October 2021, viewed 8 November 2026. https://arxiv.org/abs/2102.01951

16. Chronologically Consistent Large Language Models. Songrun He, Linying Lv, Asaf Manela and Jimmy Wu, arXiv 2502.21206, version 3, 6 July 2025, viewed 8 November 2026. https://arxiv.org/abs/2502.21206

17. Anonymization and Information Loss. Ke Wu, Baozhong Yang, Zhenkun Ying and Dexin Zhou, arXiv 2511.15364, version 2, 5 September 2026, viewed 8 November 2026. https://arxiv.org/abs/2511.15364

18. Learning Point: Point in Time. FISD Alternative Data Council, SIIA, undated (posted under 2021/06 on fisd.net), viewed 8 November 2026. https://fisd.net/wp-content/uploads/2021/06/Learning-Point-Point-in-Time.pdf

19. The Probability of Backtest Overfitting. David H. Bailey, Jonathan M. Borwein, Marcos López de Prado and Qiji Jim Zhu, revised version, 27 February 2015, viewed 8 November 2026. https://www.davidhbailey.com/dhbpapers/backtest-prob.pdf

20. The Deflated Sharpe Ratio: Correcting for Selection Bias, Backtest Overfitting and Non-Normality. David H. Bailey and Marcos López de Prado, Journal of Portfolio Management 40(5), 2014, version of 31 July 2014, viewed 8 November 2026. https://www.davidhbailey.com/dhbpapers/deflated-sharpe.pdf

21. ForecastBench: A Dynamic Benchmark of AI Forecasting Capabilities. Ezra Karger, Houtan Bastani, Chen Yueh-Han and others, ICLR 2025, arXiv 2409.19839, version 5, 28 February 2025, viewed 8 November 2026. https://arxiv.org/abs/2409.19839

22. Real-Time Data Set for Macroeconomists. Federal Reserve Bank of Philadelphia, viewed 8 November 2026. https://www.philadelphiafed.org/surveys-and-data/real-time-data-research/real-time-data-set-for-macroeconomists

23. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. Naman Jain, King Han, Alex Gu and others, ICLR 2025, arXiv 2403.07974, version 2, 6 June 2024, viewed 8 November 2026. https://arxiv.org/abs/2403.07974

24. Benchmark Inflation: Revealing LLM Performance Gaps Using Retro-Holdouts. Jacob Haimes, Cenny Wenner, Kunvar Thaman and others, arXiv 2410.09247, version 1, 11 October 2024, viewed 8 November 2026. https://arxiv.org/abs/2410.09247

25. Time Machine GPT. Felix Drinkall, Eghbal Rahimikia, Janet B. Pierrehumbert and Stefan Zohren, NAACL Findings 2024, arXiv 2404.18543, version 1, 29 April 2024, viewed 8 November 2026. https://arxiv.org/abs/2404.18543

26. Financial Statement and Notes Data Sets. U.S. Securities and Exchange Commission, viewed 8 November 2026. https://www.sec.gov/files/aqfsn_1.pdf

27. FineWeb dataset card. HuggingFaceFW, Hugging Face, viewed 8 November 2026. https://huggingface.co/datasets/HuggingFaceFW/fineweb

28. ForecastBench website. Forecasting Research Institute, viewed 8 November 2026. https://www.forecastbench.org/

About this report

Published by Rillor on 10 November 2026. Superintelligence (SI) tools assist research and drafting, and every figure cites its source. Rillor builds and secures SI systems, supplies the data they learn from and delivers the infrastructure they run on. We apply SI to financial and compute markets.

Cite as: Rillor. Look-ahead bias in LLM backtests and how to test for it. 10 November 2026. https://rillor.com/insights/look-ahead-bias-in-llm-backtests

Notices

Rillor is not a registered investment adviser or commodity trading advisor and does not provide investment or trading advice. Research, forecasts, data and software described on this site are for research and engineering use. Nothing here is an offer or recommendation to buy or sell any security, commodity interest or digital asset. Past or simulated results do not indicate future results.

Superintelligence (SI) outputs can be wrong. Important decisions should include human review.

Work with Rillor

Send the question you want forecast, its horizon and the data you already hold. Contact us at rillor.com/contact?type=forecasting or contact@rillor.com. More on forecasting and market intelligence: rillor.com/markets/forecasting